Methods For Image Recovery In Computational Imaging

by

Lei Xiao

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL
STUDIES
(Computer Science)

The University of British Columbia
(Vancouver)

April 2017

Abstract

As a classic topic that has been studied for decades, image restoration is still a very active research area. Developing more effective and efficient methods are highly desirable. This thesis addresses image restoration problems for applications in computational imaging including Time-of-Flight (ToF) imaging and digital photography.

While ToF cameras have shown great promise at low-cost depth imaging, they suffer from limited depth-of-field and low spatial-resolution. We develop a computational method to remove lens blur and increase image resolution of off-the-shelf ToF cameras. The method solves latent images directly from the raw sensor data as an inverse problem, and supports for future ToF cameras that use multiple frequencies, phases and exposures.

Photographs taken by hand-held cameras are likely to suffer from blur caused by camera shake during exposure. Removing such blur and recovering sharp images as a post-process is therefore critical. We develop a blind deblurring method that is purely based on stochastic random-walk optimization. This simple framework in combination with different priors produces comparable results to the much more complex state-of-the-art deblurring algorithms.

Blur causes even more serious issues for document photographs as slight blur can make Optical Character Recognition (OCR) techniques fail. We address the blind deblurring problem specifically for common document photographs. Observing that the latter are mostly composed of high-order structures, our method captures such domain property by a series of high-order filters as well as customized response functions. These parameters are trained from data by discriminative learning approach and form an end-to-end network that can efficiently and jointly estimate blur kernels and legible images.

Discriminative learning approaches achieve convincing trade-off between image quality and computational efficiency, however, they require separate training for each restoration task and problem condition, making it time-consuming and difficult to encompass all tasks and conditions during training. We combine discriminative learning and formal optimization techniques to learn image priors that require a single-pass training and share across various tasks and conditions while keeping the efficiency as previous discriminative methods. After being trained, our method can be combined with other likelihood or priors to address unseen restoration tasks or further improve the image quality.

Preface

The content of this thesis is based on the following publications. For each publication, the contributions of each author are listed.

L. Xiao, F. Heide, M. O'Toole, A. Kolb, M. B. Hullin, K. Kutulakos, and W. Heidrich. Defocus Deblurring and Superresolution for Time-of-Flight Depth Cameras. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014. W. Heidrich conceived the initial idea of tackling the multi-path mitigation problem as an inverse problem. The shallow depth-of-field limitation of ToF cameras was raised up during discussions between L. Xiao, F. Heide and M. O'Toole. L. Xiao developed the inverse problem formulation and optimization algorithm, designed the experiment setup, performed all experiments, and wrote the initial manuscript. F. Heide contributed to discussions during the course of the research, and provided the code for lens calibration. A. Kolb provided the ToF camera used in the experiments. All authors contributed to the manuscript writing.

L. Xiao, J. Gregson, F. Heide, and W. Heidrich. Stochastic Blind Motion Deblurring. IEEE Transactions on Image Processing, 24(10):3071-3085, Oct 2015. L. Xiao conceived the idea, developed the algorithm, conducted all experiments, and wrote the initial manuscript. W. Heidrich provided detailed guidance during the course of the research, and did major editing of the final manuscript. J. Gregson and F. Heide contributed to discussions during the course of the research and the manuscript writing.

L. Xiao, J. Wang, W. Heidrich, and M. Hirsch. Learning High-Order Filters for Efficient Blind Deconvolution of Document Photographs. European Conference on Computer Vision (ECCV), 2016. The problem of document photo deblurring was proposed by J. Wang. L. Xiao developed the method, conducted all experiments, and wrote the initial manuscript. J. Wang and M. Hirsch provided guidance during the course of the research and did major editing of the the final manuscript. W. Heidrich contributed to discussions during the course of the research.

L. Xiao, F. Heide, W. Heidrich, B. Schölkopf, and M. Hirsch. Learning Proximal Operators for Image Restoration. under review of ICCV 2017. L. Xiao conceived the idea, developed the method, conducted most experiments, and wrote the initial manuscript. F. Heide contributed to discussions during the course of the research, provided the Halide implementation, and did major editing of the manuscript. M. Hirsch contributed to discussions during the course of the research, and did major editing of the manuscript. All authors contributed to the manuscript writing.

Table of Contents

List of Tables

List of Figures

Glossary

ADMM Alternating Direction Method of Multipliers

ARF Active Random Field

BCCB Block Circulant matrix with Circulant Blocks

BM3D Block Matching and 3D Filtering

CGD Conjugate Gradient Descent

CNN Convolutional Neural Network

CSC Convolutional Sparse Coding

CSF Cascaded Shrinkage Fields

EFF Efficient Filter Flow

EPLL Expected Patch Log Likelihood

FoE Field-of-Experts

GD Gradient Descent

GMM Gaussian Mixture Models

HQS Half Quadratic Splitting method

KSVD K-Singular Value Decomposition

LSQ Least Squares

LSSC	Learned Simultaneous Sparse Coding
NLM	Non-Local Mean
OCR	Optical Character Recognition
opt-MRF	Optimized Markov Random Fields
PD	First Order Primal Dual method
PMD	Photonic Mixer Device
PPI	Pixel Per Inch
PSNR	Peak Signal-to-Noise Ratio
RTF	Regression Tree Field
SCD	Stochastic Coordinate Descent
SF	Shrinkage Field
SNR	Signal-to-Noise Ratio
TGV	Total Generalized Variation
ToF	Time-of-Flight
TRD	Trainable Nonlinear Reaction Diffusion
TV	Total Variation
WNNM	Weighted Nuclear Norm Minimization

Acknowledgments

First and foremost, I would like to thank my advisor Wolfgang Heidrich for his guidance and support throughout my time at University of British Columbia.

Thanks to my thesis and RPE committee members Uri Ascher, Jim Little and Robert J. Woodham for their timely feedback and advice over the years.

Thanks to all my research collaborators: Felix Heide, James Gregson, Matthew O'Toole, Matthias B. Hullin, Andreas Kolb, Kiriakos N. Kutulakos, Jue Wang, Michael Hirsch and Bernhard Schölkopf.

Dedication

To my parents and Yuna, for their love and support.

Chapter 1

Introduction

Image restoration aims at computationally enhancing the quality of images by undoing the adverse effects of image degradation such as noise and blur. As a key area of image and signal processing it is an extremely well studied problem and a plethora of methods exists, see for example [74] for a recent survey. Restoration tasks, such as denoising, deblurring, demosaicing and resolution enhancement, have to be addressed as part of most imaging and machine vision systems.

Although tremendous efforts have been dedicated to this topic, image restoration is still a very active research area due to the following major reasons. From a Bayesian perspective, solving restoration tasks as statistical estimation problems does not only require physically-motivated models for the data likelihood, but relies on effective prior knowledge on the latent images, a.k.a. priors or regularizers, as a key component. Good image priors are still an active area of investigation. While traditional methods focus on local image statistics and aim at maintaining edges such as Total Variation (TV) [88], bilateral filter [102] and anisotropic diffusion [105], more recent methods exploit the non-local statistics of images [2, 22, 25, 37, 71, 101]. In particular, the highly successful Block Matching and 3D Filtering (BM3D) method [22] searches for similar patches within the image and combines them through a collaborative filtering step.

On the other hand, efficiently solving restoration problems with the priors is important as well. Image restoration tasks are typically formulated as inverse problems, where the latent images are estimated by solving corresponding numerical optimization problems. Significant progress has been made recently in optimization algorithms. Representative techniques such as Half Quadratic Splitting method (HQS) [30], Split Bregman [35], Alternating Direction Method of Multipliers (ADMM) [79], First Order Primal Dual method (PD) [15], etc, have been widely used in image restoration and significantly improved the restoration performance both in time and quality. In spite of this, as the rise of mobile imaging systems such as smartphone cameras and autonomous vehicles, developing even more power- and time-efficient optimization techniques are still highly desirable.

Furthermore, through the successful application of machine learning and data-driven approaches, image restoration has seen revived interest and significant progress more recently. Broadly speaking, recently proposed state-of-the-art data-driven methods can be grouped into two classes: generative approaches that aim at probabilistic models of undegraded images and discriminative approaches that try to learn a direct mapping from degraded to clean images. Generative methods seek to learn probabilistic models of undegraded images. A simple, yet powerful subclass include models that approximate the sparse gradient distribution of natural images [58, 59, 68]. More expressive generative models include Field-of-Experts (FoE) [86], K-Singular Value Decomposition (KSVD) [26] and Expected Patch Log Likelihood (EPLL) [118]. Discriminative models have become increasingly popular for image restoration due to their attractive tradeoff between high image restoration quality and computational efficiency. Methods include trainable random field models such as Cascaded Shrinkage Fields (CSF) [92], Regression Tree Field (RTF) [51], Trainable Nonlinear Reaction Diffusion (TRD) [18], as well as deep convolutional networks [50] and other multi-layer perceptrons [13]. Discriminative approaches achieve great computational efficiency at run-time by defining a particular feed-forward structure whose trainable parameters are optimized for a particular task during training. Those learned parameters are then kept fixed at run-time resulting in a fixed computational cost.

This thesis presents our researches on above topics in image restoration and demonstrate applications for Time-of-Flight (ToF) imaging and digital photography, as detailed below.

1.1 Time-of-flight depth imaging

Fast and high-quality depth-sensing cameras are highly desirable in mobile robotics, human-machine interfaces, quality control and inspection, and advanced automotive applications. Among the wide variety of depth-sensing technologies available (depth from stereo, structured lighting, Lidar scanning, etc), continuous-wave ToF cameras have emerged as an efficient, low-cost, compact, and versatile depth imaging solution, such as Microsoft Kinect-2, Photonic Mixer Device (PMD), Swiss-Ranger, etc.

The active light source required to produce these ToF images presents significant drawbacks, however. To create a ToF image with high Signal-to-Noise Ratio (SNR), the light signal must be sufficiently intense to overcome sensor noise and quantization effects. Factors determining the signal strength include light source power, integration time, imaging range, and lens aperture. In practice, light power is often limited for eye safety and energy considerations, and the integration time must be short enough to allow real-time operation. Consequently, ToF systems ideally would need to use imaging optics with large numerical aperture to make better use of available light. However, this would come at a cost; large apertures have a shallow depth of field and hence introduce defocus blur in the raw ToF images. Another shortcoming is the limited spatial resolution of currently available ToF sensors. Most commercial ToF cameras have fewer than 0.04 megapixels.

Due to the non-linear image formation model of ToF cameras, such depth of field blur and low resolution present a significant problem for ToF cameras, generating artifacts such as “flying pixels” around depth discontinuities as well as loss of texture detail.

Chapter 3 addresses this problem by introducing a new computational method to simultaneously remove the defocus blur and increase the resolution of off-the-shelf ToF cameras post capture. The method directly works on the raw sensor data and solves the deblurring and superresolution problem in a principled way. Unlike previous ToF deblurring techniques, our approach applies regularizers directly to the latent intensity and depth images, and supports deblurring ToF images captured with multiple modulation frequencies, phases or exposures. This work was published in [110].

1.2 Blind deblurring of natural images

Taking photographs has become increasingly common due to the popularity of mobile cameras in recent years. However, photos taken by hand-held cameras are likely to suffer from motion blur caused by camera shake during exposure, especially in low-light environment. The blur in an image can prevent humans from resolving scene details and make machine vision algorithms (object recognition, segmentation, etc) fail. Therefore, removing such blur and recovering sharp images as a post-process is critical.

When the camera motion trajectories are pre-known, the problem to estimate the sharp latent images is called *non-blind deblurring*. However, in most practical cases these information are unknown, and need to be estimated together with the sharp images. This more challenging problem is called *blind deblurring*. Solving blind deblurring typically requires good prior knowledge of the latent image and blur. While good priors for both the images and the blur kernels are still an active area of investigation, many choices that have been proposed are non-linear and often even non-convex. This makes it difficult and time consuming to experiment with different image priors, since each new candidate typically requires a customized optimization procedure that can require a significant effort to implement.

Chapter 4 presents a blind deblurring method that is purely based on simple stochastic sampling. The method relies entirely on local evaluations of the objective function, without the need to compute gradient information. This makes it effortless to implement and test new image and kernel priors. We demonstrate such stochastic optimization solver for a variety of non-convex and non-smooth priors and likelihood. In combination with different image and kernel priors, the method produces results that match or exceed the results obtained by much more complex state-of-the-art algorithms, which typically require customized optimization solvers. This work was published in [112].

1.3 Blind deblurring of document photographs

Taking photographs of text documents (printed articles, receipts, newspapers, books, etc) instead of scanning them has become quite common recently. The motion blur caused by camera shake is critical for document photographs taken by hand-held cameras, as slight blur can prevent existing Optical Character Recognition (OCR) techniques from extracting correct text from them. Removing blur and recovering sharp, legible document images is thus highly desirable.

Recent text image deblurring methods use sparse gradient priors (e.g., total variation [16], ℓ_0 gradient [19, 78]) and text-specific priors (e.g., text classifier [19], ℓ_0 intensity [78]) for sharp latent image estimation. These methods can produce high-quality results in many cases, however their practical adaptation is hampered by several drawbacks. Firstly, their use of sparse gradient priors usually forces the recovered image to be piece-wise constant. Although these priors are effective for images with large-font text (i.e., high Pixel Per Inch (PPI)), they do not work well for photographs of common text documents such as printed articles and newspapers where the font sizes are typically small [48]. Furthermore, these methods employ iterative sparse optimization techniques that are usually time-consuming for high resolution images taken by modern cameras.

Chapter 5 presents a new algorithm for practical document deblurring that achieves both high quality and high efficiency. Observing that document images are usually dominated by small-scale high-order structures, the algorithm learns a series of scale- and iteration-wise high-order filters to capture the domain-specific property of document photographs. The method uses a discriminative learning approach to train such filters and other parameters of a feed-forward network that takes a single blurry document photograph as input and produces high-quality latent image and blur kernel rapidly. This work was published in [111].

1.4 Learning proximal operators for general image restoration

State-of-the-art models such as FoE [86], EPLL [118], Weighted Nuclear Norm Minimization (WNNM) [37] etc, are generic in the sense that they can be applied for various restoration tasks. However, the resulting iterative optimization problems are prohibitively expensive, rendering them impractical for applications that require real-time processes and for use on mobile vision systems. Recently, a number of works [18, 87, 92] have addressed this issue by truncating the iterative optimization and learning discriminative image priors, tailored to the likelihood and optimization approach. While these methods allow one to trade-off quality with the computational budget for a given application, the learned priors are highly specialized to the image formation and noise parameters, in contrast to optimization-based approaches. Since each individual problem instantiation requires costly learning and storing of the model coefficients, current proposals for learned priors are impractical for vision applications with dynamically changing (often continuous) parameters. This is a common scenario in most real-world vision settings, as well as applications in engineering and scientific imaging that rely on the ability rapidly prototype methods.

Chapter 6 presents an algorithm that combines discriminative learned models with formal optimization methods to learn generic priors that truly share across problem domains. Using proximal optimization methods [10, 30, 79] allows us to decouple the likelihood and prior which is key to learning such shared models. It also means that we can rely on well-researched physically-motivated models for the likelihood, while learning priors from example data. By learning generalized proximal mappings as a prior model, our approach is computationally cheap while being general. We verify our technique using the same model for a variety of diverse low-level image reconstruction tasks and problem conditions, demonstrating the effectiveness of our approach. Benefiting from the proximal splitting techniques, our approach can naturally be combined with existing state-of-the-art priors after being trained to further improve the reconstruction quality. This work was published in [109].

1.5 Organization

The remaining part of this thesis is organized as follows: Chapter 2 reviews background knowledge and previous work; Chapter 3, 4, 5 and 6 present our methods and results for each individual topic; and Chapter 7 concludes this thesis work and discusses potential research directions for future work.

Chapter 2

Background and Related Work

Let vector $\mathbf{b}$ be the observed image, vector $\mathbf{x}$ be the latent (desired) image and function f be the sensing operator, the image formation process is represented as in Eq. 2.1.

$$\mathbf{b} = f(\mathbf{x}) \tag{2.1}$$

The sensing operator f differs for each imaging task. For denoising task, $f(\mathbf{x}) = \mathbf{x} + \mathbf{n}$ where $\mathbf{n}$ is the noise. For inpainting and demosaicing tasks, $f(\mathbf{x}) = \mathbf{a} \odot \mathbf{x} + \mathbf{n}$ where vector $\mathbf{a}$ is a binary mask representing the measurement pattern, and $\odot$ represents element-wise multiplication. For non-blind deconvolution tasks, $f(\mathbf{x}) = \mathbf{k} \otimes \mathbf{x} + \mathbf{n}$ where vector $\mathbf{k}$ is the blur kernel, and $\otimes$ represents two-dimensional (2D) convolution between $\mathbf{x}$ and $\mathbf{k}$. The sensing operators f are linear for these tasks, thus the image formation process can be re-written as:

$$\mathbf{b} = \mathbf{A}\mathbf{x} + \mathbf{n} \tag{2.2}$$

where matrix $\mathbf{A}$ represents the sensing operation. Specifically, for denoising task $\mathbf{A}$ is an identity matrix, for inpainting and demosaicing $\mathbf{A}$ is a binary diagonal matrix with diagonal element $\mathbf{a}$, and for non-blind deconvolution task $\mathbf{A}$ is a Block Circulant matrix with Circulant Blocks (BCCB) that represents 2D convolution with $\mathbf{k}$. For other imaging tasks such as time-of-flight depth imaging and phase retrieval, the sensing operators f are non-linear and more complex. The time-of-flight depth imaging task is discussed in Section 2.4.

Image restoration tasks are typically formulated as inverse problems in advanced methods. As the original problems are typically ill-posed, prior knowledge of the latent images are required as regularization. The general form of such regularized inverse problems is given in Eq. 2.3.

$$\mathbf{x} = \underset{\mathbf{x}}{\operatorname{argmin}} \frac{\lambda}{2} ||\mathbf{b} - \mathbf{A}\mathbf{x}||_2^2 + \mathcal{R}(\mathbf{x}), \tag{2.3}$$

where the first squared term in the objective is the data fidelity (likelihood), $\mathcal{R}(\mathbf{x})$ is the image prior, and λ is a positive scalar controlling the relative weight between the data fidelity and prior in the objective. The well-known TV prior [88], for example, is formulated as $\mathcal{R}(\mathbf{x}) = ||\nabla\mathbf{x}||_1$ where ∇ computes the derivatives of image $\mathbf{x}$, and the ℓ_1 norm promotes sparsity of $\nabla\mathbf{x}$.

The remaining part of this chapter is organized as follows: Section 2.1 reviews previous work on image restoration in general; Section 2.2 and 2.3 review previous work on blind deconvolution of natural images and document images respectively; Section 2.4 reviews related work on time-of-flight deblurring and enhancement; and Section 2.5 reviews several advanced numerical optimization methods that are frequently used in literature for image restoration.

2.1 General image restoration

Broadly speaking, recently proposed state-of-the-art image restoration methods can be grouped into three classes: *classical* approaches that make no explicit use of machine learning, *generative* approaches that aim at probabilistic models of undegraded natural images and *discriminative* approaches that try to learn a direct mapping from degraded to latent images. Unlike classical methods, methods belonging to the latter two classes depend on the availability of training data. We review each class in the following sections.

2.1.1 Classical models

Traditional models focus on local image statistics and aim at maintaining edges such as TV [88], anisotropic diffusion models [105], and bilateral filter [102]. More recent methods exploit the non-local statistics of images such as Non-Local Mean (NLM) [2], BM3D [22], non-local variants of sparse representation methods [25, 71], WNNM [37], and [101].

In particular, the seminal work of bilateral filter [102] and NLM [2] can be analyzed within the same scheme for image filtering that is based on pixel similarity, as given in Eq.2.4.

$$\begin{aligned} \hat{\mathbf{x}}(i) &= \frac{1}{C_i} \sum_{j \in \Omega_i} w(i,j) \cdot \mathbf{x}(j) \\ s.t., C_i &= \sum_{j \in \Omega_i} w(i,j), \end{aligned} \tag{2.4}$$

where $\hat{\mathbf{x}}$ is the estimated (filtered) image from input $\mathbf{x}$, i, j are pixel indices, Ω_i represents the neighbor region of pixel $\mathbf{x}(i)$, scalar weight $w(i,j)$ represents the contribution of pixel $\mathbf{x}(j)$ to the estimated pixel $\hat{\mathbf{x}}(i)$, and scalar C_i is the normalization weight. The weight $w(i,j)$ is defined on the similarity between pixel $\mathbf{x}(i)$ and $\mathbf{x}(j)$, that is, the more similar pixel $\mathbf{x}(j)$ is to pixel $\mathbf{x}(i)$, the more contribution is from pixel j when estimating $\hat{\mathbf{x}}(i)$.

With this scheme, we give the representation of Gaussian filter, bilateral filter and NLM in Eq.2.5,2.6 and 2.7 respectively.

$$\text{(Gaussian filter)} \quad \hat{\mathbf{x}}(i) = \frac{1}{C_i} \sum_{j \in \Omega_i} e^{-\frac{|i-j|^2}{\sigma^2}} \cdot \mathbf{x}(j) \tag{2.5}$$

$$\text{(bilateral filter)} \quad \hat{\mathbf{x}}(i) = \frac{1}{C_i} \sum_{j \in \Omega_i} e^{-\frac{|i-j|^2}{\sigma_v^2} - \frac{|\mathbf{x}(i)-\mathbf{x}(j)|^2}{\sigma_s^2}} \cdot \mathbf{x}(j) \tag{2.6}$$

$$\text{(non-local mean)} \quad \hat{\mathbf{x}}(i) = \frac{1}{C_i} \sum_{j \in \mathbf{x}} e^{-\frac{||\alpha(\mathbf{x}(\mathcal{N}_i)-\mathbf{x}(\mathcal{N}_j))||_2^2}{\sigma^2}} \cdot \mathbf{x}(j) \tag{2.7}$$

The Gaussian filter (Eq.2.5) defines the pixel similarity on their spatial distance $|i-j|^2$. The bilateral filter (Eq.2.6) defines the pixel similarity on both the spatial distance and the intensity difference, thus can preserve edges while reducing noise in the image. NLM (Eq.2.7) defines the similarity between pixels i, j on the similarity between patches $\mathbf{x}(\mathcal{N}_i)$, $\mathbf{x}(\mathcal{N}_j)$ centered at $\mathbf{x}(i)$ and $\mathbf{x}(j)$ respectively, and the pixel $\mathbf{x}(j)$ can be from any region in the image $\mathbf{x}$ rather than only from the local neighborhood of pixel i as in previous methods. α in Eq.2.7 is a pre-defined Gaussian kernel that gives more weight at central regions of each patch. NLM builds on the fundamental observation that similar patches often can be found within an image, and this idea has stimulated many follow-up extensions.

BM3D method extends the non-local similarity idea which searches for similar patches across the image as in NLM, but combines them through collaborative patch-filtering steps rather than simple pixel averaging. The non-local sparse representation methods [25, 71] explores the patch similarity idea as well while enforces the similar patches to have similar coefficients in the transformed domain. WNNM [37] filters similar patches within the image by applying low-rank constraints on singular value decomposition.

2.1.2 Generative learning models

Methods of this class seek to learn probabilistic models of undegraded natural images. A simple yet powerful subclass include models that approximate the sparse gradient distribution of natural images [58, 59, 68]. The ℓ_p-norm ($0 < p < 1$) on image derivative has been shown as an effective prior to enforce the heavy-tailed distribution of the gradient of natural images [58, 68].

More expressive generative models include KSVD [26], Convolutional Sparse Coding (CSC) [12, 41, 108], FoE [86] and EPLL [118]. Both the KSVD and CSC methods assume small patches in an image can be approximated by a linear combination of a few atoms from an overcomplete dictionary $\mathbf{D}$ that is learned from training data. The dictionary $\mathbf{D}$ consists of K atoms $\{\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_K\}$, $||\mathbf{d}_k||_2^2 \leq 1$. The problem KSVD aims to solve is given in Eq.2.8.

$$\boldsymbol{\alpha} = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}}\, ||\boldsymbol{\alpha}||_0, \quad \text{s.t.} \quad \mathbf{x} = \sum_{k=1}^{K} \alpha_k \mathbf{d}_k, \tag{2.8}$$

where $\mathbf{x}$ is a small patch from the image, and $\boldsymbol{\alpha} = \{\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, ..., \boldsymbol{\alpha}_K\}$ are the coefficients. The ℓ_0-norm enforces the coefficients $\boldsymbol{\alpha}$ to be sparse. As KSVD operates on each image patch separately, it ignores the coherence between patches across the image, and the resulting dictionary $\mathbf{D}$ typically presents redundancy. To address this issue, CSC methods [12, 41, 108] are recently proposed where the dictionary and coefficients are learned on the entire images rather than patches. The problem that CSC methods solve is given in Eq.2.9.

$$\boldsymbol{\alpha} = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} ||\mathbf{x} - \sum_{k=1}^{K} \mathbf{d}_k \otimes \boldsymbol{\alpha}_k||_2^2 + \lambda ||\boldsymbol{\alpha}||_1, \tag{2.9}$$

where the image $\mathbf{x}$ is approximated by a linear combination of K atoms $\mathbf{d}_k$ each convolved with a sparse coefficient map $\boldsymbol{\alpha}_k$. Each $\boldsymbol{\alpha}_k$ has the same support as the image $\mathbf{x}$.

KSVD and CSC are categorized as synthesis-operator methods. The counterpart are the analysis-operator methods, of which a representative work is the FoE model as given in Eq.2.10.

$$\mathbf{x} = \underset{\mathbf{x}}{\operatorname{argmin}} \frac{\lambda}{2} ||\mathbf{b} - \mathbf{A}\mathbf{x}||_2^2 + \sum_{i=1}^{N} \phi_i(\mathbf{F}_i \mathbf{x}), \tag{2.10}$$

where matrix $\mathbf{F}_i$ represents 2D convolution with filter $\mathbf{f}_i$, and function ϕ_i represents the penalty on corresponding filter response $\mathbf{F}_i\mathbf{x}$. Both the filters $\mathbf{f}_i$ and penalty functions ϕ_i are learned from training data. The well-known TV regularizer can be viewed as a special case of the FoE model where $\mathbf{f}_i$ is the derivative operator ∇ and ϕ_i the ℓ_1 norm.

EPLL [118] models image patches through Gaussian Mixture Models (GMM) and applies such patch prior to the whole image by HQS optimization technique. All of these methods have in common to be agnostic to the image restoration task, i.e., they can be used for any image degradation and can be combined with any likelihood and additional priors at test time.

2.1.3 Discriminative learning models

Discriminative models have become increasingly popular for image restoration recently due to their attractive tradeoff between high image restoration quality and efficiency at test time. Discriminative approaches owe their computational efficiency at run-time by defining a particular feed-forward structure whose trainable parameters are optimized for a particular task during training. Those learned parameters are then kept fixed at run-time resulting in a fixed computational cost.

Representative methods in this class include trainable random field models such as RTF [51], CSF [92], TRD [18], as well as deep convolutional networks [50] and other multi-layer perceptrons [13]. Both CSF and TRD are derived from the FoE model (given in Eq.2.10) by unrolling corresponding optimization iterations of Eq.2.10 to be feed-forward networks. All parameters of the network are trained by minimizing the error between its output images and ground truth images. Specifically, CSF and TRD unroll HQS and Gradient Descent (GD) iterations that solve Eq.2.10, respectively. More details of these methods can be found in Chapter 5 and Chapter 6.

The downside of discriminative models is that they cannot generalize across tasks and typically necessitate separate feed-forward architectures and separate training for each restoration task (denoising, demosaicing, deblurring, etc) and every possible image degradation (noise level, Bayer pattern, blur kernel, etc). In Chapter 6, we try to address this issue.

2.2 Natural image deblurring

In this section we review previous deblurring work on natural images. As most blind deblurring methods alternatively estimate the blur kernel and the sharp latent image as non-blind subproblems, we first review the state-of-the-art non-blind deblurring methods in Section 2.2.1 before the blind methods in Section 2.2.2. Although the non-blind deblurring is typically an important component of the blind methods, note that, the priors and optimization methods developed for high-quality non-blind deblurring may not be effective for the blind case.

2.2.1 Non-blind deconvolution

The classic Wiener deconvolution method [107] uses an inverse filter in Fourier domain by assuming the image and noise power spectra are known, and the results typically suffer from overly smoothed edges and ringing artifacts. The TV prior, which promotes piece-wise constant images, has become very popular in imaging problems since it was first introduced in [88]. The heavy-tailed distribution of natural image gradients is another effective regularization for image deblurring [27, 65]. Several non-local priors were developed to model the patch recurrence across an image and showed significant improvement on image deblurring [21, 23, 24, 54]. [94] first removes the blurriness in the image with a simple constrained least squares minimization and then trains a neural network to remove the noise and ringing artifacts in the result image from the previous step. This method needs to train separate neural network for each blur kernel, which limits its application in practice. More recently, several methods were proposed to use trainable random field models for image restoration including deblurring [92, 93]. The CSF method [92] reduces the optimization problem of random field models into cascaded quadratic minimization problems that can be efficiently solved in Fourier domain.

2.2.2 Blind deconvolution

When the blur kernel is not readily available, the problem becomes much more challenging especially for single image input, where both the latent sharp image $\mathbf{x}$ and blur kernel $\mathbf{k}$ need to be recovered, as shown in Eq. 2.11.

$$(\mathbf{x}, \mathbf{k}) = \operatorname*{argmin}_{\mathbf{x},\mathbf{k}} \frac{\lambda}{2}||\mathbf{b} - \mathbf{k} \otimes \mathbf{x}||_2^2 + \mathcal{R}(\mathbf{x}) + \mathcal{T}(\mathbf{k}), \tag{2.11}$$

where $\mathcal{T}(\mathbf{k})$ represents priors on the blur kernel. Typically $\mathbf{k}$ is assumed to be non-negative and sum up to one, i.e., $\mathbf{k} \geq 0$ and $||\mathbf{k}||_1 = 1$. This is an ill-posed problem to solve. It is highly under-constrained and non-convex, and is subject to many degenerate solutions, including one where the estimated kernel is simply a Dirac peak and the latent image is the blurry input.

In order to make the non-convex optimization converge to a good local optimum, the regularizers should be discriminatively designed for the blind problem, while directly applying the image priors from non-blind deblurring research (e.g., TV) usually works poorly especially when k is large or complex. Another, the designed regularizers should be efficiently solvable by the optimization method for practical use.

Recent success arises from the use of sparse priors and multi-scale scheme. Fergus *et al* [27] fits the heavy-tailed prior by a mixture of Gaussians and solves the intrinsic image gradient and blur kernel by a variational Bayesian method [75].

Richardson-Lucy algorithm [70, 84] is then applied to reconstruct the final intrinsic image with the estimated kernel. Krishnan *et al* [59] introduced a scale-invariant ℓ_1/ℓ_2 prior, which compensates for the attenuation of high frequencies in the blurry image. Xu *et al* [114] used the ℓ_0 regularizer on the image gradient. Goldstein *et al* [34] estimated the kernel from the power spectrum of the blurred image. Yue *et al* [115] improved [34] by fusing it with sparse gradient prior. Sun *et al* [99] imposed patch priors to recover good partial latent images for kernel estimation. Michaeli and Irani [73] exploited the recurrence of small image patches across different scales of single natural images. Anwar *et al* [5] learned a class-specific prior of image frequency spectrum for the restoration of frequencies that cannot be recovered with generic priors. Zuo *et al* [119] learned iteration-wise parameters of the ℓ_p regularizer on image gradients. Schelten *et al* [90] trained cascaded interleaved RTF [93] to post-improve the result of other blind deblurring methods for natural images.

Another type of methods employs filtering approaches to extract strong image edges from which kernels may be estimated rapidly. Cho *et al* [20] adopted shock filter [77] and bilateral filter [102] to predict sharp edges. Xu *et al* [113] improved [20] by neglecting edges with small spatial support as they impede kernel estimation. Schuler *et al* [95] learned such nonlinear filters with a multi-layer convolutional neural network.

In case of highly noisy input, Tai *et al* [100] proposed a method for jointly denoising and deblurring the image. Zhong *et al* [116] applied directional low-pass filters at different orientations to the input image and estimated the Radon transform of the blur kernel from each filtered image, while the final blur kernel is computed by inverse Radon transform.

For non-uniform blur due to camera rotation, Whyte *et al* [106] proposed a parameterized geometric model of the blurring process considering the rotational velocity of the camera during exposure. Gupta *et al* [38] modeled the spatially varying kernels by a motion density function which records the fraction of time spent in each discretized portion of the space by the camera during exposure. Harmeling *et al* [40] and Hirsch *et al* [46] combined the global camera motion model and local patch uniform deblurring to accelerate the non-uniform kernel estimation.

2.3 Document photograph deblurring

In this section we review previous deblurring work on text document photographs. Most recent methods of text deblurring use the same sparse gradient assumption developed for natural images, and augment it with additional text-specific regularization. Chen *et al* [16] and Cho *et al* [19] applied explicit text pixel segmentation and enforced the text pixels to be dark or have similar colors. Pan *et al* [78] used ℓ_0 regularized intensity and gradient priors for text deblurring. The use of sparse gradient priors makes such methods work well for large-font text images, but fail on common document images that have smaller fonts.

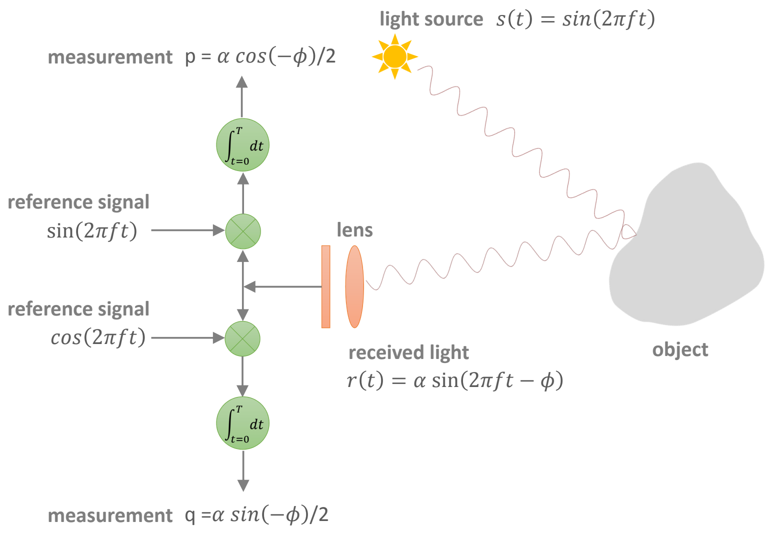
Hradiš *et al* [48] trained a convolutional neural network to directly predict the sharp patch from a small blurry one, without considering the image formation model and explicit blur kernel estimation. With a large enough model and training dataset, this method produces good results on English documents with severe noise, large defocus blurs or simple motion blur. However, this method fails on more complicated motion trajectories, and is sensitive to page orientation, font style and text languages. Furthermore, this method often produces "hallucinated" characters or words that appear to be sharp and natural in the output image, but are completely wrong semantically. This undesirable side-effect severely limits its application range as most users do not expect the text to be changed in the deblurring process.

2.4 Time-of-flight imaging

2.4.1 Imaging principle

We explain the principle of ToF depth imaging with Fig.2.1. The active light source emits modulated light wave with frequency f, i.e., $s(t) = \sin(2\pi f t)$, where t denotes time. The light reflected from the object surface with reflectance α falls on the sensor with a phase delay ϕ, i.e., $r(t) = \alpha \sin(2\pi f t - \phi)$. The phase delay ϕ relates to the depth z of the surface point by Eq.2.12:

$$z = \frac{c\phi}{4\pi f} \tag{2.12}$$



light source $s(t) = sin(2\pi ft)$
measurement p = $\alpha\ cos(-\phi)/2$
$\int_{t=0}^{T} dt$
reference signal
$\sin(2\pi ft)$
lens
reference signal
$cos(2\pi ft)$
received light
$r(t) = \alpha \sin(2\pi ft - \phi)$
object
$\int_{t=0}^{T} dt$
measurement q =$\alpha\ sin(-\phi)/2$

Figure 2.1: Principle of ToF imaging.

where c is the light speed. In the sensor, the received signal $r(t)$ is further modulated with two reference signals $\sin(2\pi ft)$ and $\cos(2\pi ft)$, and the resulting modulated signals are integrated over time duration $0 - T$ to generate the ToF measurements $p = \alpha\cos(-\phi)/2$ and $q = \alpha\sin(-\phi)/2$. With these two measurements, the phase ϕ and the reflectance α can be calculated as:

$$\phi = \arctan(-\frac{q}{p}) \tag{2.13}$$

$$\alpha = 2\sqrt{p^2 + q^2}, \tag{2.14}$$

while the depth of the object surface point can be calculated by Eq.2.12. Note that by Euler's formula, the two ToF measurements p, q can be viewed as the real and imaginary part of the complex value $\alpha e^{-i\phi}/2$, that is, $ae^{-i(\frac{4\pi f}{c}\cdot z)}$, where $a = \alpha/2$.

The complex measurements for all sensor pixels are represented as

$$\mathbf{a} \circ e^{-i(\frac{4\pi f}{c} \cdot \mathbf{z})}, \tag{2.15}$$

where $\mathbf{a}$ and $\mathbf{z}$ represent the amplitude and depth map respectively, and $\circ$ represents Hadamard product (pixel-wise multiplication) of vectors.

2.4.2 ToF defocus deblurring

Most existing ToF enhancement methods take as input the naive depth map from the camera software, rather than the raw complex-valued measurements. Lidar-Boost [96] and KinectFusion [49] utilized captures from multiple viewpoints to increase the depth resolution. Zhu *et al* [117] explored the complementary characteristics of ToF and stereo geometry methods and combined them to produce better-quality depths. Other methods ([28, 53, 80]) used a high-resolution RGB or intensity image to guide the upsampling of the low-resolution depth map, based on the assumption about the co-occurrence of image discontinuities in RGB and depth data. These methods assume the scenes are all in-focus, while in this paper we deal with the scenes degraded by defocus blur. Furthermore, these methods require additional hardware or multi-view captures, while our method uses single-view captures from a single ToF sensor.

Godbaz *et al* [32] proposed a two-stage method for parametric blind deconvolution of full-field continuous-wave Lidar imaging. They estimate the lens parameters from a pair of Lidar measurements taken at different aperture settings, and then deconvolve these complex-domain measurements, from which the final depth map is computed. Godbaz *et al* [31] applied the coded aperture technique to extend the depth of field for full-field continuous-wave Lidar imaging. The complex-domain Lidar measurement is iteratively deconvolved with a simple Gaussian derivative prior, while at each iteration the blur kernel of each pixel is updated according to the currently estimated Lidar image. These two methods are close to ours in the sense of directly working on the raw measurements. In contrast to these methods, which aim to deblur the complex measurements, our approach directly estimates the latent amplitude and depth from the degraded measurements. This allows us to apply separate regularizations on the amplitude and depth, and also supports for the next generation ToF cameras with multiple modulation frequencies, phases and exposures [33, 55].

Single-frequency ToF cameras have limited unambiguous distance range. Objects separated by the integer multiples of the full range are indistinguishable. The next generation of ToF cameras use multiple modulation frequencies and phases to reduce the ambiguity [33, 55]. The multi-frequency/phase data can also help resolve "flying pixels" (mixtures of foreground and background depth) at the object boundaries, and suppress artifacts due to global illumination [29]. The ToF data captured with single exposure could be noisy or saturated due to scene properties such as surface materials and reflectivity. Multiple exposures are proposed to increase the dynamic range of the measurements and remove those unreliable pixels [33, 39]. Our algorithm adapts well to these cameras, since it directly estimates the latent amplitude and depth from raw measurements that could come from multiple sequential captures.

2.5 Proximal optimization techniques

Numerical optimization methods play important roles in image restoration. Recently, a class of algorithms, called *proximal algorithms*, have attracted widespread interest in computational imaging and image processing as they are very generally applicable and well-suited to large-scale problems [79]. In proximal algorithms, the basic operation is evaluating the *proximal operator* of a function, which involves a small optimization problem and possibly has a closed-form solution. In the remaining part of this section, we present the definition of the proximal operators and review two specific proximal algorithms (HQS, ADMM) that are widely used in recent work.

2.5.1 Proximal operators

Given function $f : \mathbb{R}^n \to \mathbb{R}$, its proximal operator is defined as

$$\mathbf{prox}_{f/\lambda}(\mathbf{v}) = \underset{\mathbf{x}}{\operatorname{argmin}}\, f(\mathbf{x}) + \frac{\lambda}{2}||\mathbf{x} - \mathbf{v}||_2^2 \tag{2.16}$$

$\mathbf{prox}_{f/\lambda}(\mathbf{v})$ can be interpreted as a point in domain $\mathbb{R}^n$ that compromises between minimizing the function f and being close to point $\mathbf{v}$. The positive scalar λ can be interpreted as the trade-off parameter. More details of proximal operators can be found in [79]. Below we present several representative functions and corresponding proximal operators.

When f is an indicator function,

$$f = \mathcal{I}_{\mathcal{C}}(\mathbf{x}) = \begin{cases} 0, \mathbf{x} \in \mathcal{C} \\ +\infty, \mathbf{x} \notin \mathcal{C} \end{cases}, \quad \mathbf{prox}_{f/\lambda}(\mathbf{v}) = \underset{\mathbf{x} \in \mathcal{C}}{\operatorname{argmin}} \|\mathbf{x} - \mathbf{v}\|_2^2 \tag{2.17}$$

When f is the ℓ_0 norm, its proximal operator is called *hard-shrinkage*:

$$f = \|\mathbf{x}\|_0, \quad \mathbf{prox}_{f/\lambda}(\mathbf{v}) = \begin{cases} 0, & |\mathbf{v}| \leq 1/\lambda \\ \mathbf{v}, \text{otherwise} \end{cases} \tag{2.18}$$

When f is the ℓ_1 norm, its proximal operator is called *soft-shrinkage*:

$$f = \|\mathbf{x}\|_1, \quad \mathbf{prox}_{f/\lambda}(\mathbf{v}) = \begin{cases} \mathbf{v} + 1/\lambda, & \mathbf{v} < -1/\lambda \\ 0, & |\mathbf{v}| \leq 1/\lambda \\ \mathbf{v} - 1/\lambda, & \mathbf{v} > 1/\lambda \end{cases} \tag{2.19}$$

2.5.2 Half quadratic splitting

In this section, we explain the HQS algorithm. For simplicity purposes, we take the widely used TV prior as an example. Eq. 2.20 gives the objective function to minimize.

$$\frac{\lambda}{2}\|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2 + \|\nabla \mathbf{x}\|_1 \tag{2.20}$$

The objective function in Eq. 2.20 is non-smooth and difficult to solve with traditional optimization methods such as Newton's method. HQS [30] relaxes the original objective in Eq. 2.20 to be:

$$\frac{\lambda}{2}\|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2 + \|\mathbf{z}\|_1 + \rho\|\mathbf{z} - \nabla \mathbf{x}\|_2^2 \tag{2.21}$$

where a slack variable $\mathbf{z}$ is introduced to approximate $\mathbf{x}$, and ρ is a positive scalar. Eq. 2.21 is iteratively minimized by solving the latent image $\mathbf{x}$ and the slack variable $\mathbf{z}$ alternately, as given in Eq. 2.22 and 2.23.

$$\mathbf{x}^t = \underset{\mathbf{x}}{\operatorname{argmin}}\, \lambda||\mathbf{b} - \mathbf{A}\mathbf{x}||_2^2 + \rho^t||\mathbf{z}^{t-1} - \nabla\mathbf{x}||_2^2 \tag{2.22}$$

$$\mathbf{z}^t = \underset{\mathbf{z}}{\operatorname{argmin}}\, ||\mathbf{z}||_1 + \rho^t||\mathbf{z} - \nabla\mathbf{x}^t||_2^2 \tag{2.23}$$

Importantly, ρ^t increases as the iteration t continues. The latter forces $\mathbf{z}$ to become an increasingly good approximation of $\mathbf{x}$, thus making Eq. 2.21 an increasingly good proxy for Eq. 2.20.

The latent image $\mathbf{x}$ update step in Eq. 2.22 is a linear least squares problem, which can be efficiently solved by Conjugate Gradient Descent (CGD), or has closed-form solution in Fourier domain when the sensing matrix $\mathbf{A}$ is identity at denoising task or BCCB at deconvolution task as given in Eq. 2.24.

$$\mathbf{x}^t = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(\lambda\mathbf{A}^\mathsf{T}\mathbf{b} + \rho^t\nabla^\mathsf{T}\mathbf{z}^t)}{\mathcal{F}(\lambda\mathbf{A}^\mathsf{T}\mathbf{A} + \rho^t\nabla^\mathsf{T}\nabla)}\right), \tag{2.24}$$

where $\mathcal{F}$ and $\mathcal{F}^{-1}$ represent Fourier and inverse Fourier transform respectively.

The slack variable $\mathbf{z}$ update step in Eq. 2.23 is pixel-wise separable and has a closed-form solution with the proximal operator *soft-shrinkage*:

$$\mathbf{z}^t = \text{soft-shrinkage}(\nabla\mathbf{x}^t, 0.5/\rho^t) = \begin{cases} \nabla\mathbf{x}^t + 0.5/\rho^t, & \nabla\mathbf{x}^t < -0.5/\rho^t \\ 0, & |\nabla\mathbf{x}^t| \leq 0.5/\rho^t \\ \nabla\mathbf{x}^t - 0.5/\rho^t, & \nabla\mathbf{x}^t > 0.5/\rho^t \end{cases} \tag{2.25}$$

The soft-shrinkage operator is introduced in Eq.2.19.

HQS typically requires only a few iterations to converge with proper setting of ρ^t in practice.

2.5.3 Alternating direction method of multipliers

ADMM [9, 79] relaxes the original objective in Eq. 2.20 to be:

$$\mathbf{x} = \underset{\mathbf{x}}{\operatorname{argmin}} \frac{\lambda}{2}||\mathbf{b} - \mathbf{A}\mathbf{x}||_2^2 + ||\mathbf{z}||_1 + \rho||\mathbf{z} - \nabla\mathbf{x} + \mathbf{u}||_2^2 \tag{2.26}$$

where slack variables $\mathbf{z}$ and $\mathbf{u}$ are introduced. Eq. 2.26 is iteratively minimized by solving for the latent image $\mathbf{x}$, slack variable $\mathbf{z}$ and $\mathbf{u}$ alternately, as given in Eq. 2.27, 2.28 and 2.29.

$$\mathbf{x}^t = \underset{\mathbf{x}}{\operatorname{argmin}}\, \lambda||\mathbf{b} - \mathbf{A}\mathbf{x}||_2^2 + \rho||\mathbf{z}^{t-1} - \nabla\mathbf{x} + \mathbf{u}^{t-1}||_2^2 \tag{2.27}$$

$$\mathbf{z}^t = \underset{\mathbf{z}}{\operatorname{argmin}}\, ||\mathbf{z}||_1 + \rho||\mathbf{z} - \nabla\mathbf{x}^t + \mathbf{u}^{t-1}||_2^2 \tag{2.28}$$

$$\mathbf{u}^t = \mathbf{u}^{t-1} + \mathbf{z}^t - \nabla\mathbf{x}^t \tag{2.29}$$

where in contrast to HQS method, the scalar ρ does not need to be increased with the iteration t. ADMM typically requires tens of iterations to converge in practice.

Chapter 3

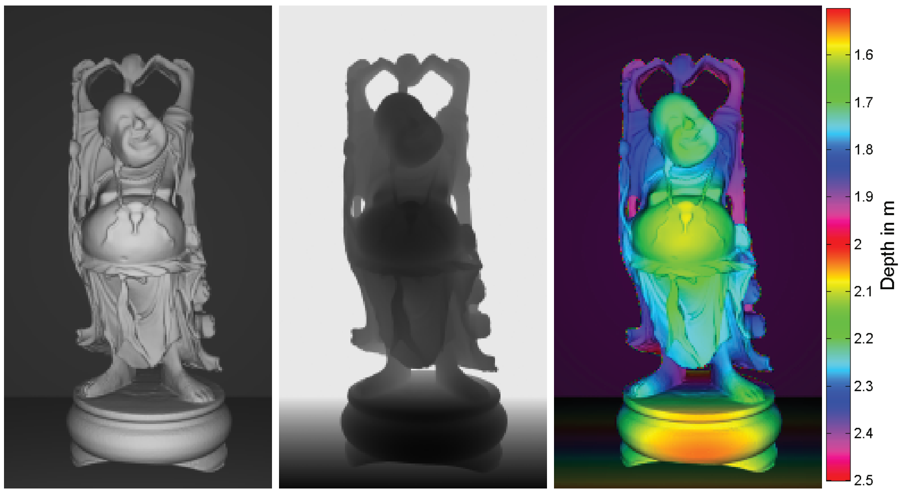
Time-of-Flight Defocus Deblurring and Superresolution

3.1 Introduction

Continuous-wave ToF cameras show great promise as low-cost depth image sensors in mobile applications. However, they also suffer from several challenges, including limited illumination intensity, which mandates the use of large numerical aperture lenses, and thus results in a shallow depth of field, making it difficult to capture scenes with large variations in depth. Another shortcoming is the limited spatial resolution of currently available ToF sensors.

In this chapter, we address this problem by introducing a new computational method to simultaneously remove defocus blur and increase the resolution of off-the-shelf ToF cameras. We do this by solving a semi-blind deconvolution problem, where prior knowledge of the blur kernel is available. Unlike past ToF deblurring techniques, our approach applies sparse regularizers directly to the latent amplitude and depth images, and supports deblurring ToF images captured with multiple frequencies, phases or exposures.

Continuous-wave ToF sensors are designed to have an image formation model that is linear in amplitude $\mathbf{a}$, but non-linear in depth $\mathbf{z}$, such that the captured raw



(a) Amplitude $\mathbf{a}$ **(b)** Depth $\mathbf{z}$ **(c)** HSV map

Figure 3.1: Simulated Buddha scene. The HSV map uses amplitude $\mathbf{a}$ as value and depth $\mathbf{z}$ as hue. The value and hue visualizes the magnitude and scaled phase of the complex-valued $\mathbf{a} \circ \mathbf{g}(\mathbf{z})$ in Eq.3.1, respectively.

sensor data is given as

$$\mathbf{a} \circ \mathbf{g}(\mathbf{z}) \approx \mathbf{a} \circ e^{-i(\frac{4\pi f}{c} \cdot \mathbf{z})}, \tag{3.1}$$

where $\circ$ represents Hadamard product, f represents the frequency of the continuous-wave modulation, and c is the constant speed of light. The function $\mathbf{g}(\mathbf{z})$ can either be calibrated [42], or, more commonly, is simply approximated by the complex-valued function from Eq. 3.1 ("cosine model" [33]). The derivation of the image formation model is given in Section 2.4. Fig. 3.1 shows a simulated scene with visualization.

We aim to compute a solution to the following ill-posed inverse problem introduced in [31]:

$$\mathbf{b} = \mathbf{SK}(\mathbf{z}) \left(\mathbf{a} \circ \mathbf{g}(\mathbf{z}) \right), \tag{3.2}$$

where the complex-valued vector $\mathbf{b}$ represents the raw ToF measurements, the real-valued matrix $\mathbf{S}$ is a downsampling operator, and the real-valued matrix $\mathbf{K}(\mathbf{z})$ represents the spatially-varying blur kernel for a given depth map $\mathbf{z}$. The problem is ill-posed because the matrix $\mathbf{SK}(\mathbf{z})$ is usually not invertible, and semi-blind because the matrix $\mathbf{K}(\mathbf{z})$ is known at each depth $\mathbf{z}$. In past work, $\mathbf{S}$ is assumed to be the identity matrix.

It is important to note that Eq.3.2 becomes the conventional image deblurring problem when $f = 0$, such that $\mathbf{a} \circ \mathbf{g}(\mathbf{z}) = \mathbf{a}$. Estimating the amount of defocus blur from a blurry amplitude map $\mathbf{a}$ is a particularly challenging problem in this case, requiring either multiple photos or specialized optics [68]. Unlike conventional cameras, ToF cameras provide additional depth information that can be used to recover the defocus blur kernel much more robustly [31].

Our method focuses on solving Eq.3.2 to recover the deblurred amplitude and depth maps from a single blurry image, captured with an off-the-shelf ToF camera. Because this inverse problem is still an ill-conditioned problem, it's critical to choose appropriate regularizers to reflect prior information on the solution (i.e., sparse edges). Godbaz *et al* [31] proposed differential priors that operate on the complex ToF image representing the cosine model, but it remains unclear what a good regularizer should even look like in this space. We instead choose to regularize our solution in the amplitude and depth map space directly. This introduces certain numerical challenges, because of the highly nonlinear relation between the depth components and the raw ToF measurements (Eq.3.1). We relax this problem by splitting the optimization procedure into two parts, alternating between optimizing for amplitude and depth. Our method can seamlessly include a super-resolution component, helping to overcome the limited sensor resolution in current generation ToF cameras. Unlike earlier approaches, our method is also not inherently limited to the cosine model, and could be easily extended to calibrated waveforms in the future.

3.2 Method

3.2.1 Algorithm overview

Given the raw measurements $\mathbf{b}$ from a single view, our algorithm aims to remove optical lens blur and produce high quality depth map $\mathbf{z}$ and amplitude map $\mathbf{a}$. The latent depth $\mathbf{z}$ and amplitude $\mathbf{a}$ are coupled in the measurements, thus we solve them as a joint optimization problem.

$$(\mathbf{a}, \mathbf{z}) = \underset{\mathbf{a},\mathbf{z}}{\text{argmin}} \, ||\mathbf{b} - \mathbf{SK}\left(\mathbf{a} \circ \mathbf{g}(\mathbf{z})\right)||_2^2 + \mathbf{\Phi}(\mathbf{a}) + \mathbf{\Psi}(\mathbf{z}) \tag{3.3}$$

Eq.3.3 shows the objective function we aim to minimize. The quadratic term represents a data-fitting error, assuming zero-mean Gaussian noise in the measurements. $\mathbf{\Phi}(\mathbf{a})$ and $\mathbf{\Psi}(\mathbf{z})$ represent regularizers for amplitude $\mathbf{a}$ and depth $\mathbf{z}$ respectively. The algorithm alternatively estimates $\mathbf{a}$ and $\mathbf{z}$, and update the blur kernel matrix $\mathbf{K}$ at the end of each iteration according to currently estimated $\mathbf{z}$.

Sparse gradient priors [68] have been widely used in natural image deblurring, but are improper for depth where the gradient could be non-zero for most pixels. The sparse second-order derivative priors have been used in surface denoising [6, 7, 104], but fail to model the discontinuity at object boundaries and thus do not distinguish the blurred and latent sharp depth. In this paper, we use the second-order Total Generalized Variation (TGV) [11] for both the amplitude and depth, as shown in Eq.3.4 and3.5.

$$\mathbf{\Phi}(\mathbf{a}) = \min_{\mathbf{y}} \lambda_1 ||\nabla \mathbf{a} - \mathbf{y}||_1 + \lambda_2 ||\nabla \mathbf{y}||_1 \tag{3.4}$$

$$\mathbf{\Psi}(\mathbf{z}) = \min_{\mathbf{x}} \tau_1 ||\nabla \mathbf{z} - \mathbf{x}||_1 + \tau_2 ||\nabla \mathbf{x}||_1 \tag{3.5}$$

The TGV prior automatically balances the first and second order derivative constraints. Following Knoll *et al* [56], this can be intuitively understood as follows. In flat regions of $\mathbf{z}$, the second order derivative $\nabla^2 \mathbf{z}$ is locally small, thus it benefits the minimization problem in Eq.3.5 to choose $\mathbf{x} = \nabla \mathbf{z}$, and minimize the second order derivative $||\nabla \mathbf{x}||_1$. While in the sharp edges of $\mathbf{z}$ (i.e., at object boundaries), $\nabla^2 \mathbf{z}$ is larger than $\nabla \mathbf{z}$, thus it benefits to choose $\mathbf{x}$ as zero, and minimize the first order derivative $||\nabla \mathbf{z}||_1$. Similar analysis applies for $\mathbf{a}$ as well. The parameters $\lambda_1, \lambda_2, \tau_1, \tau_2$ define the relative weights of the first and second order constraints. A modified version of TGV was used in Ferstl *et al* [28] for image

Algorithm 1 Defocus deblurring for ToF depth camera

```
Input: Raw measurements $\mathbf{b}$; modulation frequency $f$; upsampling ratio $r$
Output: Estimated depth $\mathbf{z}$ and amplitude $\mathbf{a}$
1: $\mathbf{a} = \text{upsample}(\text{magnitude}(\mathbf{b}), r)$
2: $\mathbf{z} = \text{upsample}(\text{phase}(\mathbf{b})/(4\pi f/c)), r)$
3: for $n = 1$ to $N$ do
4:     $\mathbf{K} = \text{updateKernel}(\mathbf{z})$
5:     $\mathbf{c} = \underset{\mathbf{c}}{\text{argmin}}\, ||\mathbf{b} - \mathbf{SKc}||_2^2 + \rho||\mathbf{c} - \mathbf{a} \circ \mathbf{g}(\mathbf{z})||_2^2$
6:     $\mathbf{a} = \underset{\mathbf{a}}{\text{argmin}}\, \rho||\mathbf{c} - \mathbf{a} \circ \mathbf{g}(\mathbf{z})||_2^2 + \mathbf{\Phi}(\mathbf{a})$
7:     $\mathbf{z} = \underset{\mathbf{z}}{\text{argmin}}\, \rho||\mathbf{c} - \mathbf{a} \circ \mathbf{g}(\mathbf{z})||_2^2 + \mathbf{\Psi}(\mathbf{z})$
8: end for
```

guided depth upsampling.

$\mathbf{a} \circ \mathbf{g}(\mathbf{z})$ in Eq.3.1 is highly nonlinear regarding to $\mathbf{z}$. To reduce the computation complexity in this nonlinear problem, the algorithm splits the data-fitting term in the objective (Eq.3.3) into a linear Least Squares (LSQ) and a pixel-wise separable nonlinear LSQ term, as in Eq.3.6. The scalar ρ defines the relative weight of the splitting term.

$$(\mathbf{a}, \mathbf{z}) = \underset{\mathbf{a},\mathbf{z},\mathbf{c}}{\text{argmin}}\, \overbrace{||\mathbf{b} - \mathbf{SKc}||_2^2}^{\text{linear LSQ for } \mathbf{c}} + \overbrace{\rho||\mathbf{c} - \mathbf{a} \circ \mathbf{g}(\mathbf{z})||_2^2}^{\text{separable nonlinear LSQ for } \mathbf{z}} + \mathbf{\Phi}(\mathbf{a}) + \mathbf{\Psi}(\mathbf{z}) \tag{3.6}$$

Algorithm 1 shows the high-level pseudocode of the proposed method. The amplitude $\mathbf{a}$ and depth $\mathbf{z}$ are initialized as the magnitude and phase of the complex-valued measurement $\mathbf{b}$, respectively, and upsampled by nearest-neighbor method if superresolution wanted. Then the algorithm iteratively updates the blur kernel matrix $\mathbf{K}$, slack variable $\mathbf{c}$, amplitude $\mathbf{a}$ and depth $\mathbf{z}$. The number of iterations N is typically set as 10. Details of each subproblem are described in Section 3.2.2 - Section 3.2.5.

3.2.2 Updating kernel estimate

The blur kernel is pre-calibrated at each pixel and sampled depth. The algorithm updates $\mathbf{K}$ by a simple interpolated lookup in the pre-calibrated table of kernels according to the currently estimated depth $\mathbf{z}$. The details of the calibration procedure are explained in Section 3.2.6.

3.2.3 Updating slack variable

The update of the slack variable $\mathbf{c}$ requires solving a linear LSQ problem (Algorithm 1, Line 5). Since the resulting linear equation system is positive-definite, a number of options for efficient solvers exist.

3.2.4 Update amplitude estimate

Algorithm 2 Update amplitude

```
Input: $\mathbf{c}$, $\mathbf{A}$, $\rho$, $\rho_a$, $\lambda_1$, $\lambda_2$, number of iterations: $M$
Output: Estimated amplitude $\mathbf{a}$
1: for $n = 1$ to $M$ do
2:     $\mathbf{a} = \underset{\mathbf{a}}{\text{argmin}}\, \rho||\mathbf{c} - \mathbf{A}\mathbf{a}||_2^2 + \lambda_1\rho_a||\nabla\mathbf{a} - \mathbf{y} - \mathbf{p}_1 + \mathbf{u}_1||_2^2$
3:     $\mathbf{y} = \underset{\mathbf{y}}{\text{argmin}}\, \lambda_1||\nabla\mathbf{a} - \mathbf{y} - \mathbf{p}_1 + \mathbf{u}_1||_2^2 +$
               $\lambda_2||\nabla\mathbf{y} - \mathbf{p}_2 + \mathbf{u}_2||_2^2$
4:     $\mathbf{p}_1 = \underset{\mathbf{p}_1}{\text{argmin}}\, ||\mathbf{p}_1||_1 + \rho_a||\nabla\mathbf{a} - \mathbf{y} - \mathbf{p}_1 + \mathbf{u}_1||_2^2$
5:     $\mathbf{p}_2 = \underset{\mathbf{p}_2}{\text{argmin}}\, ||\mathbf{p}_2||_1 + \rho_a||\nabla\mathbf{y} - \mathbf{p}_2 + \mathbf{u}_2||_2^2$
6:     $\mathbf{u}_1 = \mathbf{u}_1 + \nabla\mathbf{a} - \mathbf{y} - \mathbf{p}_1$
7:     $\mathbf{u}_2 = \mathbf{u}_2 + \nabla\mathbf{y} - \mathbf{p}_2$
8: end for
```

By substituting $\mathbf{\Phi}(\mathbf{a})$ into the update rule for the amplitude (Algorithm 1, Line 6), we obtain the following optimization problem

$$\min_{\mathbf{a},\mathbf{y}} \rho||\mathbf{c} - \mathbf{A}\mathbf{a}||_2^2 + \lambda_1||\nabla\mathbf{a} - \mathbf{y}||_1 + \lambda_2||\nabla\mathbf{y}||_1, \tag{3.7}$$

where $\mathbf{A}$ is a diagonal matrix composed of $\mathbf{g}(\mathbf{z})$. This problem is solved by ADMM [9]), as shown in Algorithm 2. The $\mathbf{a}$ and $\mathbf{y}$ updates are linear least squares problems.

The $\mathbf{p}_{1,2}$-updates are soft shrinkage problems and have closed form solutions [9]. The number of ADMM iterations M is typically set to 20. More details of each subproblem are provided in Appendix A.1.

3.2.5 Updating depth estimate

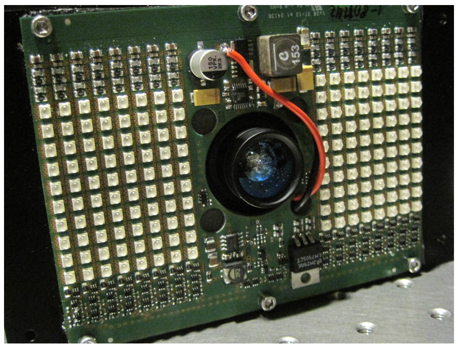
Algorithm 3 Update depth

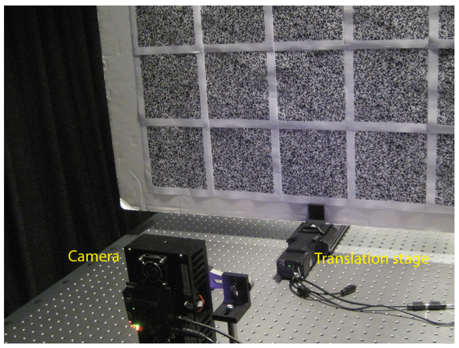
```
Input: $\mathbf{c}$, $\mathbf{a}$, $\rho$, $\rho_x$, $\tau_1$, $\tau_2$, number of iterations: $M$
Output: Updated depth $\mathbf{z}$
1: for $n = 1$ to $M$ do
2:     $\mathbf{z} = \underset{\mathbf{z}}{\operatorname{argmin}}\, \rho||\mathbf{c} - \mathbf{a} \circ \mathbf{g}(\mathbf{z})||_2^2 + \tau_1\rho_x||\nabla\mathbf{z} - \mathbf{x} - \mathbf{q}_1 + \mathbf{v}_1||_2^2$
3:     $\mathbf{x} = \underset{\mathbf{x}}{\operatorname{argmin}}\, \tau_1||\nabla\mathbf{z} - \mathbf{x} - \mathbf{q}_1 + \mathbf{v}_1||_2^2 + \tau_2||\nabla\mathbf{x} - \mathbf{q}_2 + \mathbf{v}_2||_2^2$
4:     $\mathbf{q}_1 = \underset{\mathbf{q}_1}{\operatorname{argmin}}\, ||\mathbf{q}_1||_1 + \rho_x||\nabla\mathbf{z} - \mathbf{x} - \mathbf{q}_1 + \mathbf{v}_1||_2^2$
5:     $\mathbf{q}_2 = \underset{\mathbf{q}_2}{\operatorname{argmin}}\, ||\mathbf{q}_2||_1 + \rho_x||\nabla\mathbf{x} - \mathbf{q}_2 + \mathbf{v}_2||_2^2$
6:     $\mathbf{v}_1 = \mathbf{v}_1 + \nabla\mathbf{z} - \mathbf{x} - \mathbf{q}_1$
7:     $\mathbf{v}_2 = \mathbf{v}_2 + \nabla\mathbf{x} - \mathbf{q}_2$
8: end for
```

In a similar fashion, we can substitute $\boldsymbol{\Psi}(\mathbf{z})$ into the update rule for the depth (Algorithm 1, Line 7), and obtain the optimization problem

$$\min_{\mathbf{z},\mathbf{x}} \rho||\mathbf{c} - \mathbf{a} \circ \mathbf{g}(\mathbf{z})||_2^2 + \tau_1||\nabla\mathbf{z} - \mathbf{x}||_1 + \tau_2||\nabla\mathbf{x}||_1 \tag{3.8}$$

Once again, we apply the ADMM method to reduce this problem into easier subproblems, as shown in Algorithm 3. For the sparse nonlinear least squares problem of updating $\mathbf{z}$ (Algorithm 3, Line 2), we use the Levenberg-Marquardt algorithm [61, 76] with an analytical Jacobian for the cosine model. To adapt our method to arbitrary (calibrated) waveforms, the only required change would be to replace this derivative estimate with a tabulated version based on the calibration data. We use the cosine model for the experiments in Section 3.3 for fair comparisons with previous work, which makes the same assumption. Again, the $\mathbf{q}_{1,2}$ updates are soft shrinkage problems, we use $M = 20$ iterations, and all further details are provided in Appendix A.1.


(a) PMD-Digicam camera



(b) Scene setup for PSF calibration

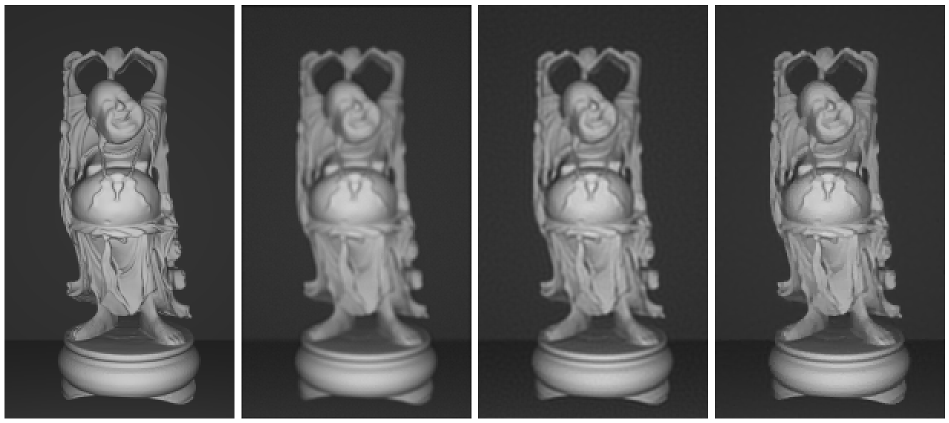
Figure 3.2: Experimental setup.

3.2.6 Calibration

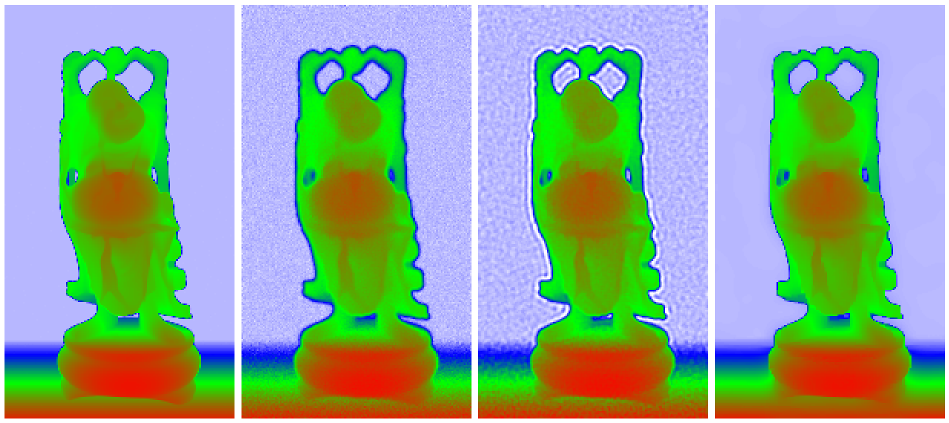
The blur kernel (PSF) pre-calibration for real measurements is done by a similar approach to [44]. Fig. 3.2 shows the experiment setup. Printed random noise patterns are attached on a flat white board, which is held on a translation stage. The translation stage moves the white board to place from 60cm to 160cm away from the camera, with 1cm incremental. At each place, the camera captures with large aperture (the same aperture used for real measurements). Then, this process is repeated but with a small aperture so that the scene is nearly in-focus. Next, the amplitude images of the two captures at each place are used to estimate the PSF as a non-blind deconvolution problem.

3.3 Results

We test the proposed algorithm on both synthetic and real datasets, and compare with two methods: the naive method, which computes the amplitude and depth as the magnitude and phase of the raw complex images respectively; and Godbaz *et al* [31], which alternatively updates blur kernels and deconvolves the complex image with a Gaussian prior.



(a) Estimated $\mathbf{a}$, *from left to right*, by ground truth, naive method (29.1dB), Godbaz method (31.9dB) and ours (34.5dB).



(b) Estimated $\mathbf{z}$, *from left to right*, by ground truth, naive method (36.2dB), Godbaz method (37.3dB) and ours (43.0dB).

Figure 3.3: Results on simulated Buddha scene with 0.5% white noise. Our method significantly reduces the blurriness and suppresses noise in both $\mathbf{a}$ and $\mathbf{z}$, and reducing the flying pixels at object boundaries. PSNR of the results are provided in the brackets.

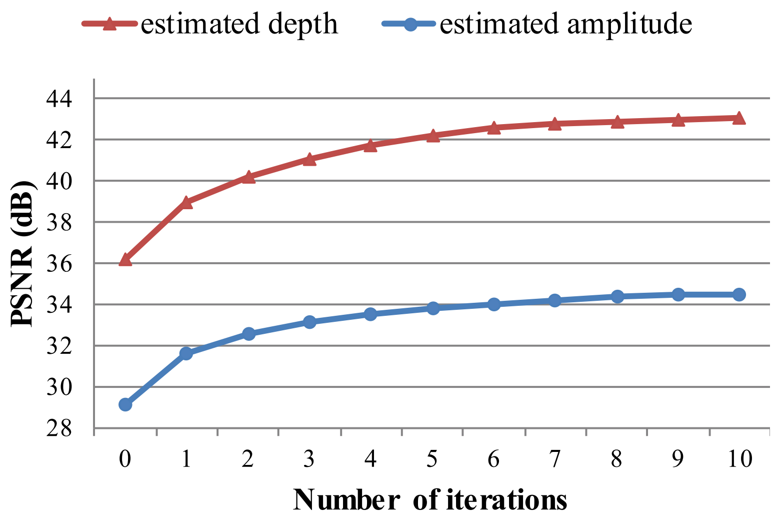


Figure 3.4: PSNR of our estimated amplitude $\mathbf{a}$ and depth $\mathbf{z}$ on the Buddha scene at each iteration.

3.3.1 Synthetic data

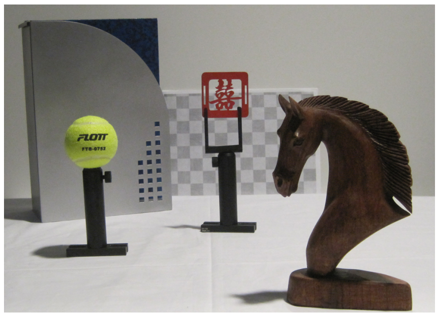
The results on a simulated Buddha scene is shown in Fig. 3.3. The naive amplitude $\mathbf{a}$ and depth $\mathbf{z}$ are blurry and contain strong noise and flying pixels. In the visualized depth map, the flying pixels appear in different color than the foreground and background surface at the boundaries. Godbaz *et al* method is highly sensitive to noise. Their result to some extent reduces the blurriness, but contains obvious noise, ringing artifacts and flying pixels. The proposed method significantly reduces the blurriness and flying pixels, and suppress the noise in both the amplitude $\mathbf{a}$ and depth $\mathbf{z}$. The results are compared with ground truth data. Our approach produces much higher PSNR than the other methods.

In Fig. 3.4, we show the PSNR values of our estimated $\mathbf{a}$ and $\mathbf{z}$ at each iteration (i.e., n in Algorithm 1).

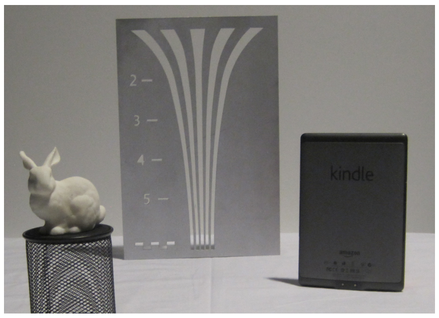
3.3.2 Real data

We captured real datasets using the Digicam camera from PMDTechnologies (Fig. 3.2) with a 6-15mm and f/1.4 lens. 30MHz modulation frequency and 300 microsecond exposure time are used, and a single frame is captured for each scene. We crop

(a) Camel scene



(b) Character scene



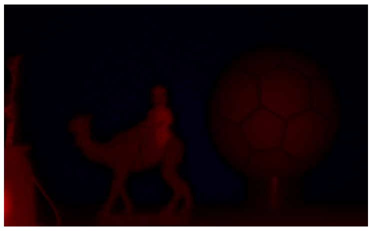
(c) Board scene

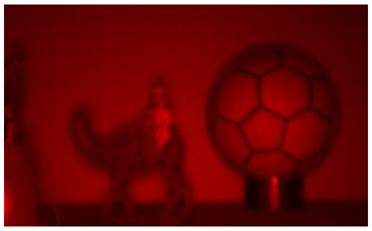
Figure 3.5: RGB Photographs of the real scenes.

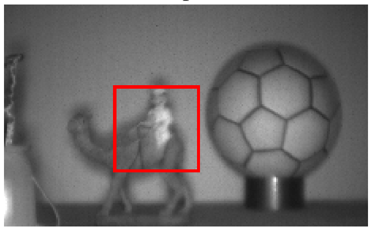
out the pixels near image boundaries and the typical resolution of input images is 250×180 pixels. The pre-calibrated PSFs have a width of 5-11 pixels at depths between 0.6 and 1.6m.

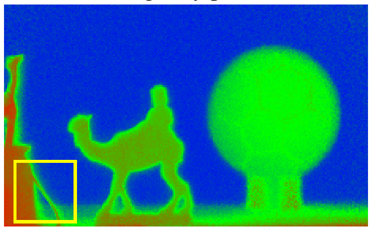
In Fig. 3.5 we show the photographs of the captured scenes. The results and comparisons are shown in Fig. 3.6, 3.7 and 3.8, and cropped regions are shown in Fig. 3.9, 3.11 and 3.12. In Fig. 3.10, we show the mesh geometry color-coded according to surface normal to better illustrate the depth results. Please zoom in for better views.

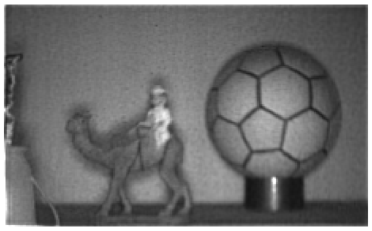
Similarly as in the synthetic example, Godbaz *et al* method is unable to handle noise (which is common in low-end ToF cameras), and fails to recover sharp scene features. Our method produces much higher quality amplitude and depth, in term of suppressing the noise, recovering sharp features and reducing flying pixels. We also run our algorithm with 2x superresolution (i.e., upsampling ratio $r = 2$ in

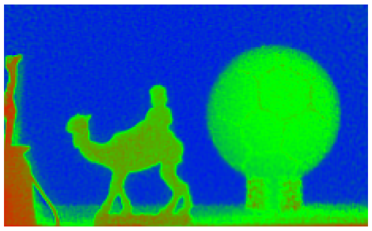

(a) Real part of **b**

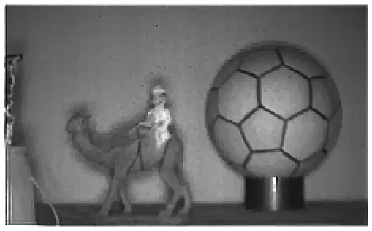

(b) Imaginary part of **b**

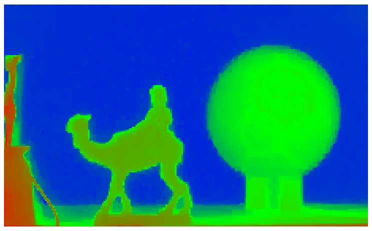

(c) Naïve**a**

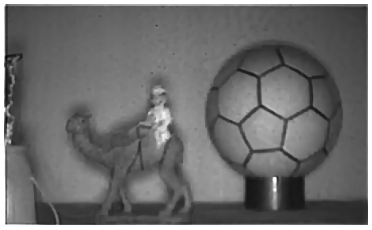

(d) Naïve**z**


(e) Godbaz **a**


(f) Godbaz **z**


(g) Our **a**


(h) Our **z**


(i) Our **a** with superresolution

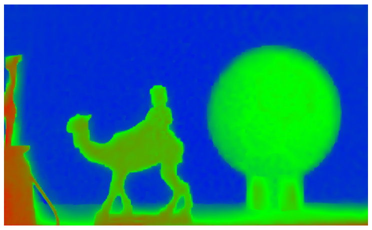
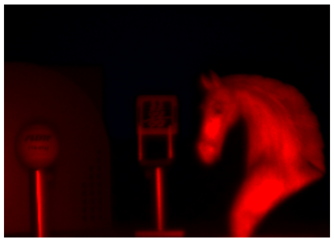
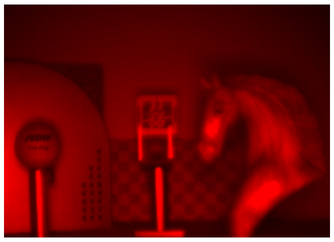

(j) Our **z** with superresolution

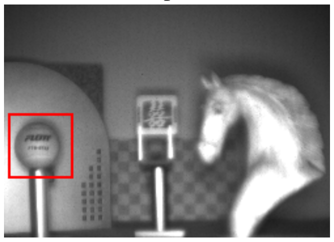
Figure 3.6: Results on the Camel scene. In (a-b) the red color indicates positive values and blue negative in the raw measurements. The cropped regions are shown in Fig. 3.9.



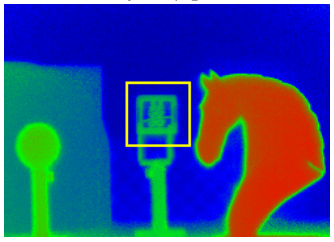
(a) Real part of **b**



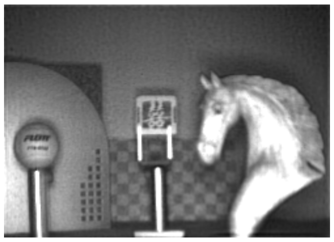
(b) Imaginary part of **b**



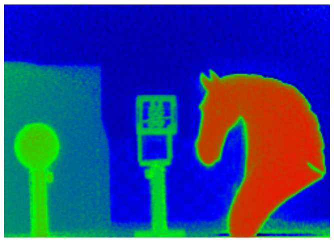
(c) Naïve**a**



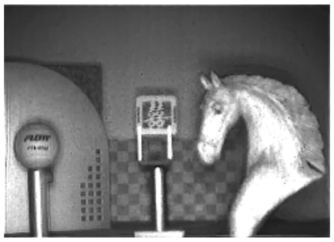
(d) Naïve**z**



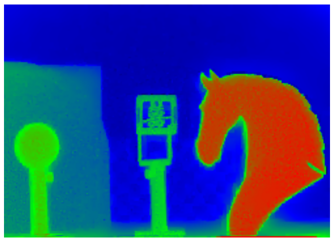
(e) Godbaz **a**



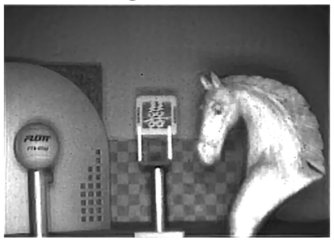
(f) Godbaz **z**



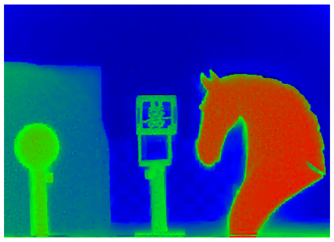
(g) Our **a**



(h) Our **z**

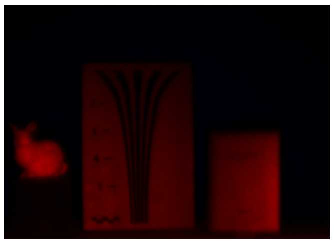


(i) Our **a** w/ superresolution

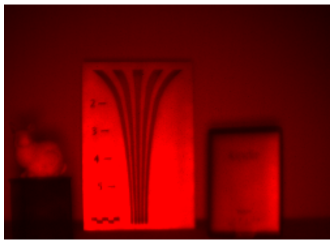


(j) Our **z** w/ superresolution

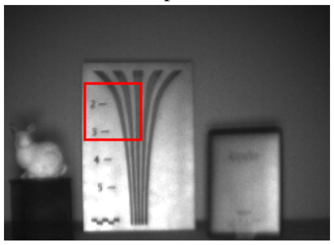
Figure 3.7: Results on the Character scene. The cropped regions are shown in Fig. 3.11.



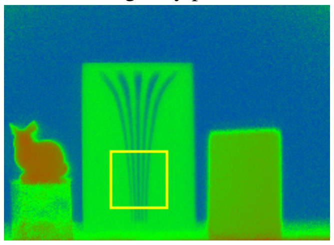
(a) Real part of **b**



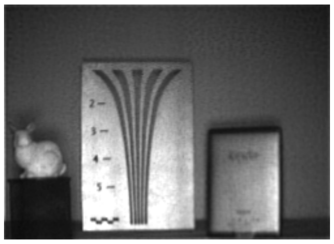
(b) Imaginary part of **b**



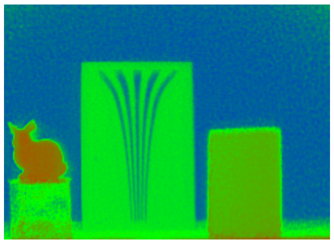
(c) Naïve**a**



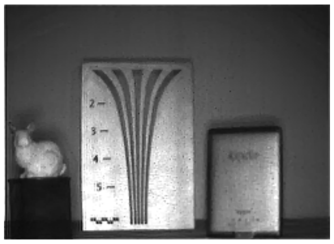
(d) Naïve**z**



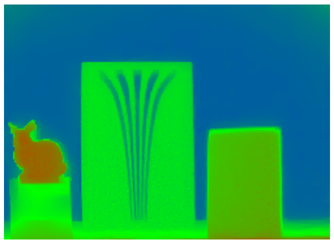
(e) Godbaz **a**



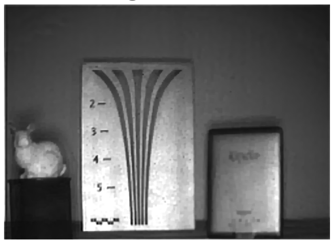
(f) Godbaz **z**



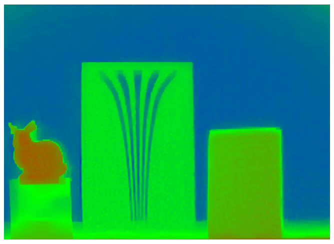
(g) Our **a**



(h) Our **z**



(i) Our **a** w/ superresolution



(j) Our **z** w/ superresolution

Figure 3.8: Results on the Board scene. The cropped regions are shown in Fig. 3.12.

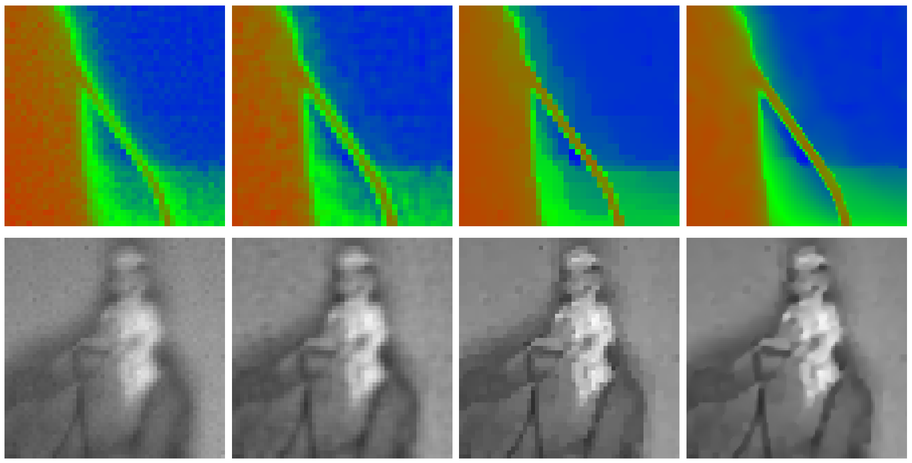


Figure 3.9: Two insets of the results on the Camel scene in Fig. 3.6. *From left to right* shows the naive method, Godbaz *et al* method, ours, and ours with superresolution.

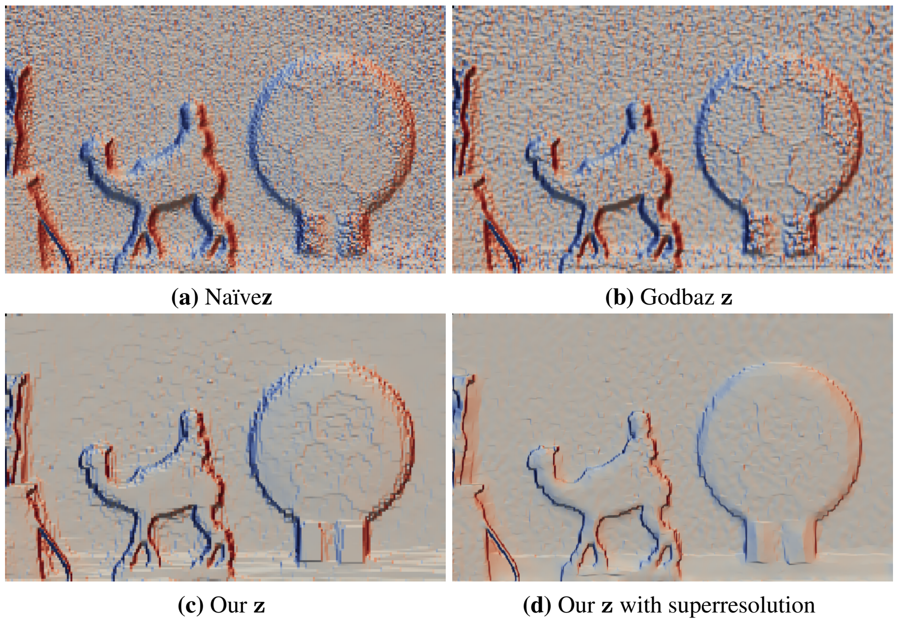


Figure 3.10: Mesh visualization for the Camel scene in Fig. 3.6. The color indicates the surface normal in horizontal direction, i.e., blue indicates left-faced surface and red the opposite.

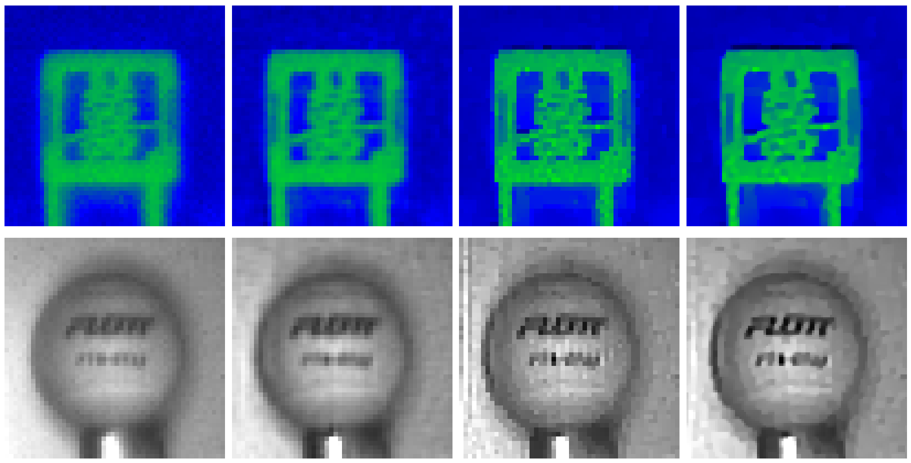


Figure 3.11: Two insets of the results on the Character scene in Fig. 3.7. *From left to right* shows the naive method, Godbaz *et al* method, ours, and ours with superresolution.

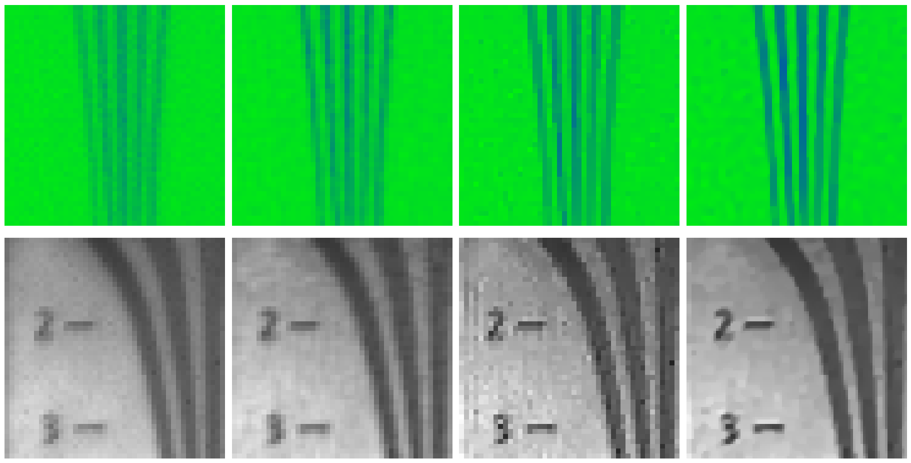


Figure 3.12: Two insets of the results on the Board scene in Fig. 3.8. *From left to right* shows the naive method, Godbaz *et al* method, ours, and ours with superresolution.

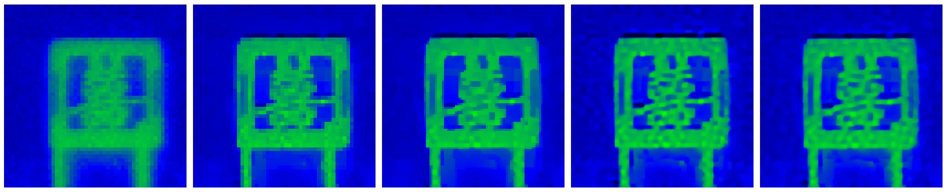


Figure 3.13: We run the proposed algorithm on an inset of the Character scene with different upsampling ratios. *From left to right* shows the result of naive method, and ours with 1x, 2x, 4x, 6x superresolution respectively. We observed that little more details are recovered beyond 4x upsampling.

Algorithm 1), and show that we achieve even better results with superresolution in our joint optimization framework. We use bicubic interpolation for the down-sampling operator $\mathbf{S}$. In Fig. 3.13, we show our results with different upsampling ratios, and observe that little more details are recovered beyond 4x upsampling for our dataset.

We tune the parameters to generate results of compared method and ours. For our results, ρ (in Algorithm 1), ρ_a (in Algorithm 2) and ρ_x (in Algorithm 3) are fixed as 0.125, 10 and 10 respectively. In Algorithm 2, λ_1 is typically set as 0.001 or 0.002, and λ_2 as 20 or 40 times λ_1. In Algorithm 3, τ_1 is typically set as 0.0005 or 0.001, and τ_2 as 20 or 40 times τ_1.

We run our unoptimized Matlab code with a single core on an Intel i7 2.4GHz CPU. The running time is reported taken the Character scene as an example. During the total 10 iterations in Algorithm 1 with no superresolution, our code took 80 seconds for updating the slack variable $\mathbf{c}$ (Sec. 3.2.3), 16 seconds for the amplitude (Sec. 3.2.4), and 347 seconds for the depth (Sec. 3.2.5). We believe the code can be further accelerated by choosing more efficient solvers for some subproblems or running on GPU. For example, the current Levenberg-Marquardt Matlab solver used in Algorithm 3 can be replaced with much more efficient ones [3].

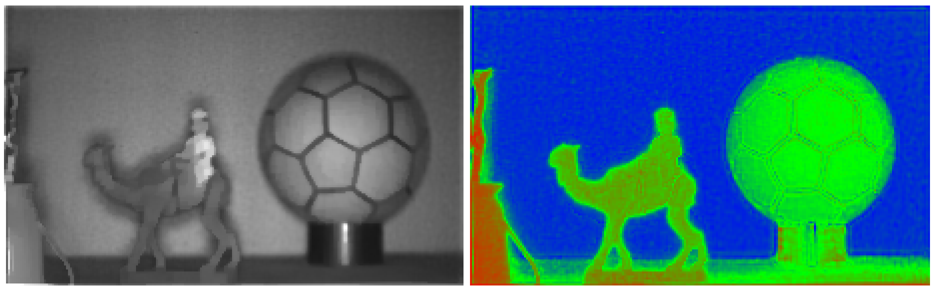

(a) Estimated $\mathbf{a}$ **(b)** Estimated $\mathbf{z}$

Figure 3.14: Experiment results using the TGV prior on the complex images. The estimated amplitude is over-smoothed while the depth is still highly noisy.

3.4 Discussion

Regularization strategy

To verify the benefit of directly regularizing and solving the latent amplitude and depth, we replaced the Gaussian derivative prior in Godbaz *et al* [31] with the TGV prior on the complex image, and solved the complex image using ADMM framework. The result on the Camel scene is shown in Fig. 3.14. Even though the prior weight is high enough to over-smooth the amplitude, the estimated depth still contains strong noise compared to Fig. 3.6. This demonstrates the high quality of our results is to a large part owed to the approach of regularizing $\mathbf{z}$ and $\mathbf{a}$ directly.

Joint deblurring and superresolution

To show the advantage of our *joint* deblurring and superresolution from ToF raw measurements, we compare with the results of superresolution *after* deblurred by each method. Two example insets are shown in Fig. 3.15. Our jointly deblurred and super-resolved depth and amplitude preserve sharp features and reduce flying pixels better than the others.

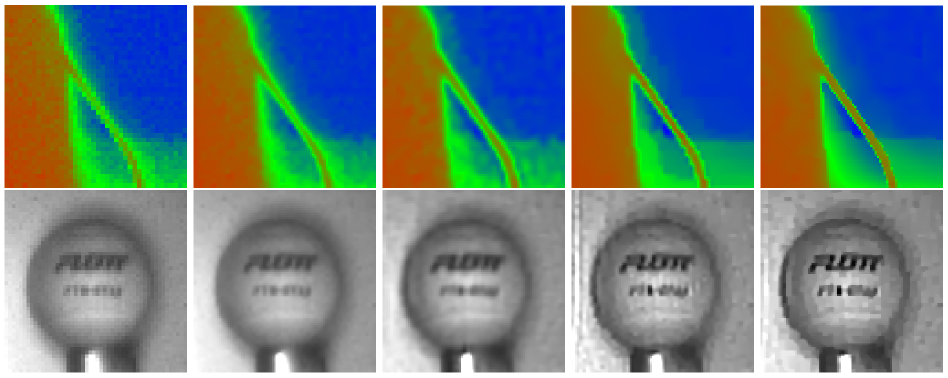


Figure 3.15: The top row shows depth results, and bottom amplitude. *From left to right*: (a) naive method; (b) naive method + post-superresolution; (c) Godbaz *et al* + post-superresolution; (d) our deblurred + post-superresolution; and (e) our jointly deblurred and superresolved. The TGV prior is used for all post-superresolution. These two scenes are from Fig. 3.9 and 3.11.

Multiple frequencies, phases and exposures

The latest generation of ToF cameras uses multiple frequencies or phases in order to reduce range ambiguity and improve depth resolution. Multiple exposures could be used to increase the dynamic range of the raw measurements and remove artifacts due to lack of reflection or over-saturation in one shot. The proposed algorithm well adapts for these cameras since the latent amplitude and depth is solved directly from the raw measurements, which could come from different captures and put together in the data-fitting term.

Defocus level

The pixel width of current ToF sensor is approximately 45μm, compared with RGB sensors which have approximately 5μm pixel size. As the ToF sensor resolution increases as the technology matures, the defocus effect in ToF imaging is expected to be more obvious and the importance of deblurring ToF images will become more pronounced.

Limitations and future work

Both the proposed method and the compared Godbaz *et al* [31] assume white Gaussian noise in the raw measurements. This noise model is inaccurate due to the non-linear image formation model, typically relatively low light conditions, as well as ambient light canceling in ToF imaging [33], which amplifies shot noise. As a future work we would like to study more accurate noise models for ToF cameras.

3.5 Conclusion

In this paper we proposed an effective method to simultaneously remove lens blur and increase image resolution for ToF depth cameras. Our algorithm solves the latent amplitude and depth directly from the raw complex images, and separate priors are used for each to recover sharp features and reduce flying pixels and noise. We show our algorithm significantly improves the image quality on simulated and real dataset compared with previous work. Unlike previous approaches, our method is not fundamentally limited to the cosine model for continuous-wave ToF cameras, which has been shown to be inaccurate for many systems (e.g., [42]) and should adapt to multi-frequency, multi-phase or multi-exposure ToF cameras.

Chapter 4

Stochastic Blind Motion Deblurring

4.1 Introduction

The goal of deconvolution is to recover a sharp intrinsic image $\mathbf{x}$ from a blurred image $\mathbf{b}$. The image formation process can be modeled as

$$\mathbf{b} = \mathbf{k} \otimes \mathbf{x} + \mathbf{n}, \tag{4.1}$$

where $\mathbf{k}$ represents the blur kernel, $\otimes$ represents discrete convolution, and $\mathbf{n}$ is a noise term. We first assume Gaussian noise for simplicity, and extend our algorithm for Poisson noise in later sections.

While there are many applications in which the kernel $\mathbf{k}$ is known or can be calibrated a priori (e.g., aberration correction) or is considered to be from a small set of possible kernels (defocus blur), in other situations the kernel is unknown since it is generated by a process that cannot be replicated (e.g., object motion or camera shake).

This blind case, where both the kernel and the intrinsic image is unknown, is highly under-constrained, and is subject to many degenerate solutions, including one where the estimated kernel is simply a Dirac peak. It is therefore obvious that blind deconvolution is only feasible with the use of additional information, either in the form of additional sensor data (e.g., inertial sensors [52] or multiple input images of the same scene [83]), or in the form of prior information (image priors). While good priors for both the images and the blur kernels are still an active area of investigation, many choices that have been proposed are non-linear and often even non-convex. This makes it difficult and time consuming to experiment with different image priors, since each new candidate typically requires a customized optimization procedure that can require a significant effort to implement.

In this chapter we extend the recent work in stochastic non-blind deconvolution [36] to derive a blind method that is purely based on stochastic sampling. The method relies entirely on local evaluations of the objective function, without the need to compute the gradient of the objective function. This makes it effortless to implement and test new image priors. Specifically we present the following contributions:

- a stochastic framework for blind deconvolution,
- a demonstration of stochastic optimization for non-convex priors for the image (e.g. sparse derivatives and cross-channel information), the kernel (e.g. anisotropic diffusion), and even the data term (handling of saturation and clamping, as well as an Anscombe transformation for Poisson noise).
- an implementation with a range of sparsity and color priors for the image, as well as sparsity and smoothness priors for the kernel, that together match or outperform existing state-of-the art methods,
- a method for recovering colored blur kernels that arrive, e.g. in remote sensing where the exposure time varies per channel (e.g., [60]), and
- a method for dealing with clipped color values in non-blind deconvolution.

4.2 Basic algorithm

Algorithm 4 Stochastic blind deconvolution framework

```
Input: Blurry image $\mathbf{b}$; weights for priors on intrinsic image: $\theta_{\mathbf{x}}$; weights for
    priors on the kernel: $\theta_{\mathbf{k}}$; maximum blur kernel size $\phi$; number of iterations $T$.
Output: Estimated intrinsic image $\mathbf{x}$; estimated kernel $\mathbf{k}$.
 1: $S = \lceil 2\log_2(\phi/3) + 1 \rceil$ //number of scales
 2: for $s = 1$ to $S$ do
 3:     $\mathbf{b}^s$ = downsample($\mathbf{b}, s$, 'bilinear')
 4:     if $s == 1$ then
 5:         $\mathbf{x}^s = \mathbf{b}^s$
 6:         $\mathbf{k}^s$ = initializeKernel($\mathbf{b}$)
 7:     else
 8:         $\mathbf{x}^s$ = upsample($\mathbf{x}^{s-1}, \sqrt{2}$, 'bicubic')
 9:         $\mathbf{k}^s$ = upsample($\mathbf{k}^{s-1}, \sqrt{2}$, 'nearest-neighbor')
10:     end if
11:     for $t = 1$ to $T$ do
12:         $\mathbf{x}^s$ = updateIntrinsicImage($\mathbf{b}^s, \mathbf{k}^s, \mathbf{x}^s, \theta_{\mathbf{x}}$)
13:         $\mathbf{k}^s$ = updateKernel($\mathbf{b}^s, \mathbf{x}^s, \mathbf{k}^s, \theta_{\mathbf{k}}$)
14:     end for
15:     $[\theta_{\mathbf{x}}, \theta_{\mathbf{k}}]$ = updateWeights($\theta_{\mathbf{x}}, \theta_{\mathbf{k}}$)
16: end for
17: $\mathbf{k} = \mathbf{k}^s$ //final estimated kernel
18: $\mathbf{x}$ = updateIntrinsicImage($\mathbf{b}, \mathbf{k}, \mathbf{x}^s, \theta_{\mathbf{x}}$) //final intrinsic image restoration
```

Our algorithm is based on a multi-scale approach that iterates between kernel estimation and solving a non-blind deconvolution step (Algorithm 4). The algorithm uses a scale space to avoid local optima, working its way from the coarse scales to the fine scales (also see Fig. 4.1). At each scale, the method upsamples the kernel and intrinsic image from the next coarser scale using nearest-neighbor and bicubic upsampling respectively and then alternately updates the current estimates for the intrinsic image (Section 4.2.1) and the kernel (Section 4.2.2) at the current scale in an inner loop. Next, prior weights are adjusted (Section 4.2.3) before moving to the next finer scale. At the coarsest scale, the kernel is initialized as a 3 by 3 image with either a horizontal or vertical stripe depending on the dominant gradient direction [27], while the blurry image for each scale is obtained with bilinear downsampling from the full-resolution blurry image.

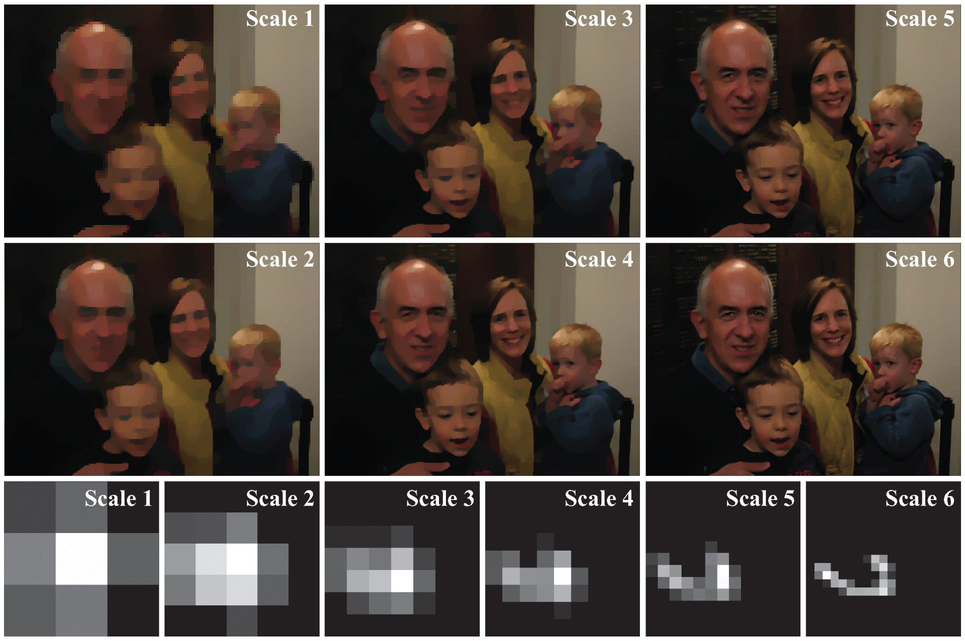


Figure 4.1: Our multi-scale scheme on the scene shown in Fig.4.8. *Row 1 & 2*: intrinsic images estimated at each scale. *Row 3*: kernel estimated at each scale. In contrast to most existing blind deconvolution methods, our algorithm can recover all color channels of the intrinsic image simultaneously using the cross-channel information.

4.2.1 Updating the intrinsic image

Given the kernel estimated at the current scale $\mathbf{k}^s$, the function updateIntrinsicImage(.) in Algorithm 4 updates our estimate of the intrinsic image by solving a non-blind deconvolution problem using a stochastic random walk optimization (Algorithm 5, also see [36]), which minimizes objectives of the form:

$$f(\mathbf{x}^s) = ||\mathbf{b}^s - \mathbf{k}^s \otimes \mathbf{x}^s||_2^2 + \theta_\mathbf{x} \cdot g(\mathbf{x}^s) \qquad (4.2)$$

The quadratic term gives the data-fitting error. $g(.) = [g_1(.), \ldots, g_R(.)]^T$ is a vector of individual regularizers, and $\theta_\mathbf{x} = [\theta_{\mathbf{x}1}, \ldots, \theta_{\mathbf{x}R}]^T$ is the corresponding vector of weights for each regularization term. The total weighted penalty (scalar) is the dot-product of $\theta_\mathbf{x}$ and $g(.)$. Examples of $g_i(.)$ are given in Eq. 4.8 - 4.12.

The intrinsic image updating method described in this section is heavily based on the previous work [36] with small modifications. We review its details here for self-completeness and provide additional analysis about its convergence.

Random walk process

As summarized in Algorithm 5, we create a random walk chain of pixel location $\mathbf{p}_i$ in the support domain of the intrinsic image at which we propose to add to or remove from an energy quantum ed_x:

$$\mathbf{x}_i^s = \mathbf{x}_{i-1}^s \pm ed_x \cdot \delta_{\mathbf{p}_i}, \tag{4.3}$$

where $\delta_{\mathbf{p}_i}$ is the characteristic function (i.e., Kronecker delta function) for the sample pixel $\mathbf{p}_i$, and $\mathbf{x}_i^s$ the estimated intrinsic image at i^{th} iteration of the random walk. Both the positive and negative energy are evaluated at the sample pixel but only the sample that decreases the objective function most is kept.

The quantity $c(\mathbf{p}_i)$ measures the change of the objective function $f(.)$ if the proposed sample $\mathbf{p}_i$ with value $\pm ed_x$ were to be accepted, i.e.,

$$c(\mathbf{p}_i) = f(\mathbf{x}_i^s) - f(\mathbf{x}_{i-1}^s) \tag{4.4}$$

If the sample decreases the objective (i.e. $c(\mathbf{p}_i) < 0$), it is accepted and Eq. 4.3 is applied (lines 6-9, Algorithm 5).

The quantity a measures the rate of objective change given the new proposed sample $\mathbf{p}_i$, as defined in Eq. 4.5.

$$a = c(\mathbf{p}_i)/c(\mathbf{p}_{i-1}) \tag{4.5}$$

If a is lower than a uniform-randomly generated real number between 0 and 1 (a.k.a.*Russian roulette* strategy in Metropolis-Hasting sampling techniques [14]), and the previous sample $\mathbf{p}_{i-1}$ decreased the objective, the new sample is discarded entirely leaving the intrinsic image and random walk chain unchanged (lines 11-13, Algorithm 5). Otherwise, the random walk chain is updated to the new sample location $\mathbf{p}_i$.

Algorithm 5 Stochastic random walk optimization

```
Input: x_0 as the initial value of the signal x to optimize, with support domain Ω;
    function c(p_i) evaluating the change of objective given sample p_i; function
    t(p_i|p_{i-1}) mutating sample location from p_{i-1} to p_i; ed as the initial mag-
    nitude of the sample energy; ε as a constant scalar (by default 0.001); γ as a
    constant scalar between 0 and 1 (by default 0.1); N as the number of iterations;
    M as the total number of proposed samples in each iteration.
Output: Updated signal x.
 1: $\mathbf{p}_0 = \text{random}(\Omega)$ //initialize the random walk chain by uniform-randomly
    sampling the support domain $\Omega$.
 2: for $j = 1$ to $N$ do
 3:     $\beta = 0$ //number of accepted samples in each iteration
 4:     for $i = 1$ to $M$ do
 5:         $\mathbf{p}_i = \text{sample}(\mathbf{p}_{i-1}, t(\mathbf{p}_i|\mathbf{p}_{i-1}), ed)$ //propose a new sample $\mathbf{p}_i$ with en-
            ergy $ed$ given the previous sample $\mathbf{p}_{i-1}$ and transition function $t(.)$.
 6:         if $c(\mathbf{p}_i) < 0$ then
 7:             $\beta = \beta + 1$
 8:             accept$(\mathbf{p}_i)$ //if the new sample $\mathbf{p}_i$ decreases the objective, accept it and
                update $\mathbf{x}$ (e.g., by Eq. 4.3).
 9:         end if
10:         $a = c(\mathbf{p}_i)/c(\mathbf{p}_{i-1})$ //compute the rate of objective change given the new
            sample $\mathbf{p}_i$.
11:         if $a <$ random$([0,1])$ and $c(\mathbf{p}_{i-1}) < 0$ then
12:             $\mathbf{p}_i = \mathbf{p}_{i-1}$ //if $a$ is lower than a uniformly distributed random real num-
                ber between 0 and 1, and the previous sample $\mathbf{p}_{i-1}$ decreased the ob-
                jective, keep the random walk chain at previous sample location $\mathbf{p}_{i-1}$;
                otherwise, update the random walk chain to $\mathbf{p}_i$.
13:         end if
14:     end for
15:     $\mathbf{p}_0 = \mathbf{p}_N$ //update the root of random walk chain for next iteration.
16:     if $\beta < \epsilon \cdot N$ and $ed < \epsilon$ then
17:         break //if too few samples were accepted and $ed$ is too small meanwhile,
            stop the iterations.
18:     else if $\beta < \gamma \cdot N$ then
19:         $ed = 0.5 \cdot ed$ //reduce the sample magnitude by half when too few samples
            were accepted.
20:     end if
21: end for
```

Evaluating $c(\mathbf{p}_i)$

Given a proposed sample, the objective function in Eq. 4.2 changes locally because of the compact support of the blur kernel $\mathbf{k}^s$ and priors $g(.)$. We can efficiently evaluate $c(\mathbf{p}_i)$ by only considering the local neighborhood of the given sample pixel.

In practice, we keep a second sequence of images $\hat{\mathbf{b}}_i^s = \mathbf{k}^s \otimes \mathbf{x}_i^s$, which represents the blurred image we would expect if the intrinsic image was $\mathbf{x}_i^s$. The $\hat{\mathbf{b}}_i^s$ can be updated efficiently by splatting $\mathbf{k}^s \otimes \delta_{\mathbf{p}_i}$ during the random walk process:

$$\hat{\mathbf{b}}_i^s = \hat{\mathbf{b}}_{i-1}^s \pm ed_x(\mathbf{k}^s \otimes \delta_{\mathbf{p}_i}) \tag{4.6}$$

The splat $\mathbf{k}^s \otimes \delta_{\mathbf{p}_i}$ is just a shifted, and mirrored copy of the kernel $\mathbf{k}^s$ and is precomputed for acceleration. The change in the regularization term is evaluated in an analogous manner but is specific to chosen regularizers.

Sample mutation

The function sample(.) generates a new sample $\mathbf{p}_i$ from the previous sample $\mathbf{p}_{i-1}$ by the mutation function $t(\mathbf{p}_i|\mathbf{p}_{i-1})$. Two types of mutation strategies are used.

The first strategy generates $\mathbf{p}_i$ by a zero-mean Gaussian-distributed random offset $\eta(0, \sigma)$ from $\mathbf{p}_{i-1}$. This mutation allows more samples to be drawn in the regions where they can reduce the objective function more effectively. Using a Gaussian distribution ensures ergodicity of the random walk process. The standard deviation of the Gaussian kernel σ is set to 4 pixels when updating intrinsic image.

The second mutation strategy chooses the new sample randomly in the support domain with uniform probability. We stimulate this mutation with 2% probability during the random walk process. This helps to avoid start-up bias and also contributes to ergodicity of the random walk process.

Eq. 4.7 gives the formula of the above mutation strategies:

$$\mathbf{p}_i = \begin{cases} random(\Omega), \text{ if } \quad q < 0.02; \\ \mathbf{p}_{i-1} + \eta(0, \sigma), \text{ otherwise} \end{cases} \tag{4.7}$$

where $random(\Omega)$ means a random pixel across the image support domain Ω, and

q is a random real number between 0 and 1 generated online at each mutation.

Sample energy

The magnitude of the sample energy ed_x is reset to be an initial large value at the beginning of each scale, and adjusted iteratively at each scale. It is reduced by half whenever the percentage of accepted samples over all proposed ones in previous iteration goes below a constant scalar $\gamma \in [0, 1]$. This allows the method to take large steps early and make more subtle changes as the minimization proceeds. In our experiments we use the initial value of ed_x as 0.02 and γ as 0.1. Note that the blurry and intrinsic images are normalized to be between 0 and 1.

Stopping criteria

The random walk process is terminated if the percentage of accepted samples in previous iteration goes below a constant threshold ϵ and the magnitude of the sample energy ed_x is smaller than ϵ meanwhile (lines 16-20, Algorithm 5). In our experiments we set ϵ to 0.001.

Regularizers $g(.)$

A benefit of the stochastic optimization framework is that it allows very general priors to be used with no change to the overall algorithm. We have used a selection of well-known and frequently used regularization terms listed below. ∇_h and ∇_v are horizontal and vertical first-order derivative operators respectively. ∇_{hh}, ∇_{hv} and ∇_{vv} are corresponding second-order derivative operators.

- Anisotropic total variation [35] (convex, but non-smooth):

$$||\nabla_h \mathbf{x}^s||_1 + ||\nabla_v \mathbf{x}^s||_1 \tag{4.8}$$

- Isotropic total variation [35] (convex, but non-smooth):

$$||(|\nabla_h \mathbf{x}^s|^2 + |\nabla_v \mathbf{x}^s|^2)^{1/2}||_1 \tag{4.9}$$

- Sparse first-order derivatives [27, 66] (non-convex):

$$||\nabla_h \mathbf{x}^s||_p + ||\nabla_v \mathbf{x}^s||_p, p \in (0, 1] \tag{4.10}$$

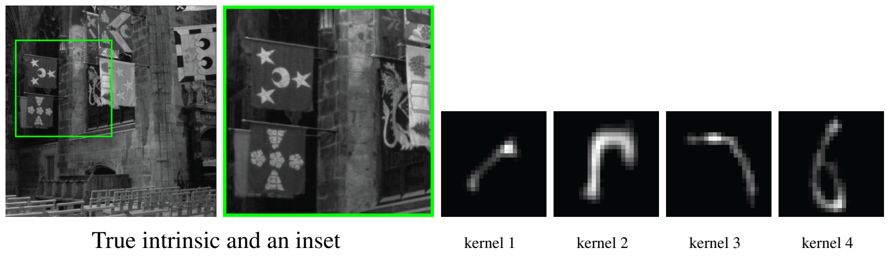


Figure 4.2: Ground truth intrinsic image (800x800 pixels) and kernels (25x25 pixels) used for empirical convergence test in Fig. 4.3 and 4.6. The kernels are upsampled by nearest neighbor for visualization.

- Sparse gradient [27, 66] (non-convex):

$$||(|\nabla_h \mathbf{x}^s|^2 + |\nabla_v \mathbf{x}^s|^2)^{1/2}||_p, p \in (0, 1] \tag{4.11}$$

- Sparse second-order derivatives [62] (non-convex):

$$||\nabla_{hh} \mathbf{x}^s||_p + 2||\nabla_{hv} \mathbf{x}^s||_p + ||\nabla_{vv} \mathbf{x}^s||_p, p \in (0, 1] \tag{4.12}$$

In addition we use other priors, including one to reduce chromatic artifacts (see Section 4.3.1) as well as non-convex data term, e.g., in the case of images with Poisson noise (Section 4.3.4).

Empirical convergence

Following the analysis in [36], our stochastic random walk framework is a form of Stochastic Coordinate Descent (SCD) [69, 89, 97]. In each iteration, the algorithm picks a single pixel in the image and checks if the objective can be reduced by depositing energy in this pixel. This corresponds to picking a single degree-of-freedom (i.e. a single coordinate axis in the vector of unknowns) and descending along that direction, without computing full gradient of the objective function. The difference to other SCD methods is that our algorithm uses the random walk process to exploit spatial coherence in the deconvolution problem and focus the computational effort on regions with sharp edges, where most work is to be done in deconvolution (see Fig. 4.4).

For smooth objectives, SCD methods provably converge as long as there is a

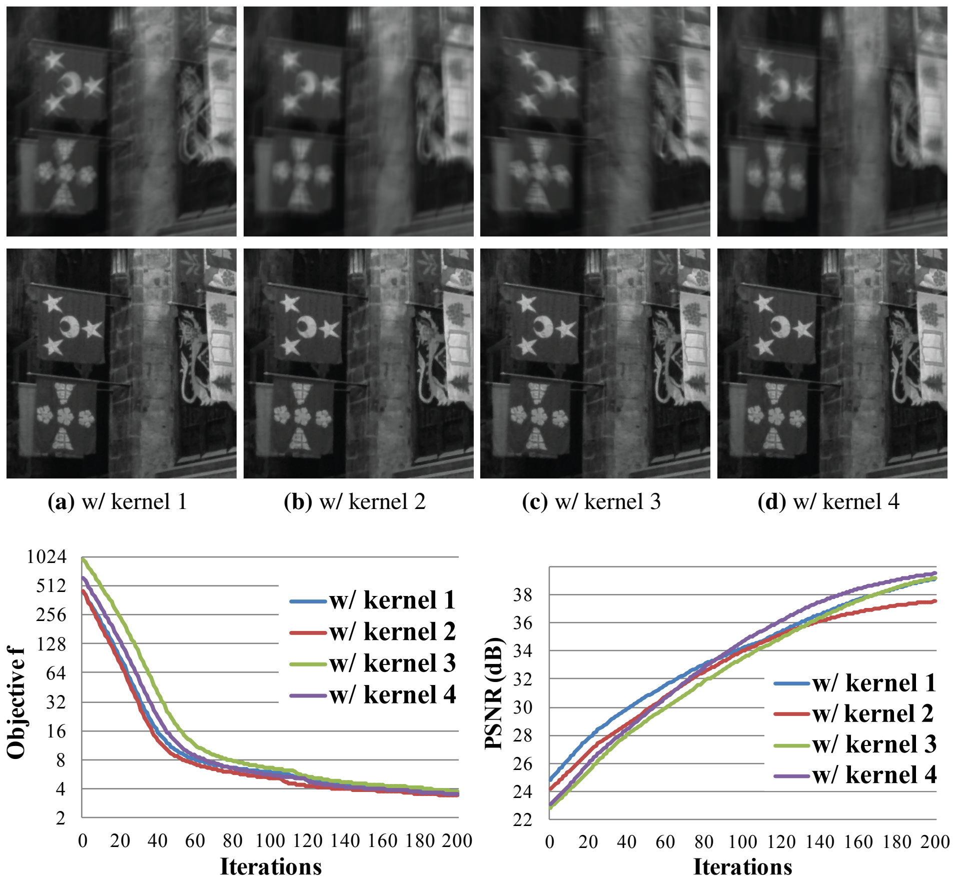
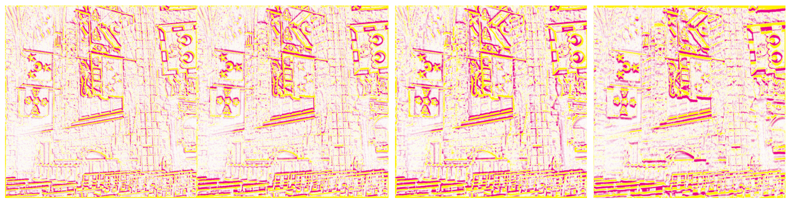
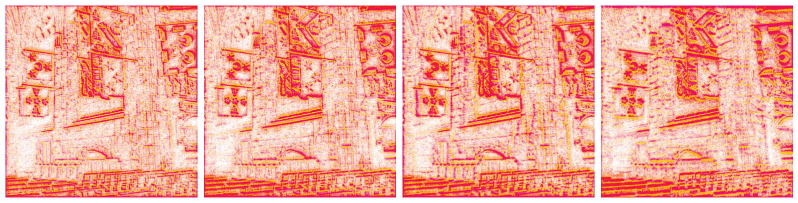


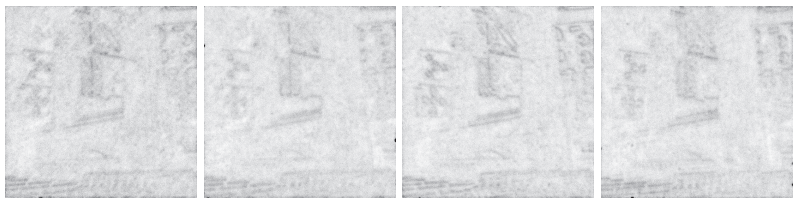
Figure 4.3: Example of *non-blind* intrinsic image estimation. In (a)-(d), the 1st row shows the inset of blurry input simulated with the true intrinsic image and each kernel given in Fig. 4.2, and the 2nd row shows the inset of non-blind estimated intrinsic image by our stochastic random walk method. Only insets are shown due to limited space. The bottom row show the change of objective function f and PSNR when the number of iterations increases. In each iteration, 50000 samples were proposed. Sparse gradient prior (Eq. 4.11 with $p = 0.8$, *non-convex*) was applied.



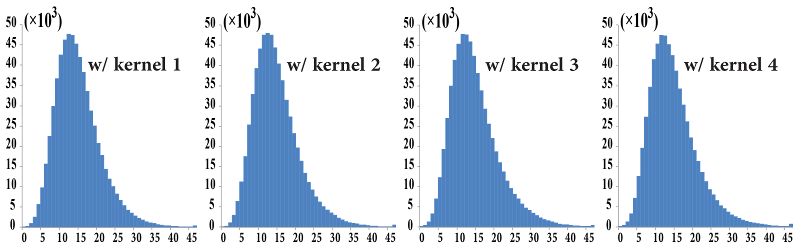
(a) Initial residual



(b) Distribution of accepted sample energy ed_x



(c) Distribution of proposed sample locations



(d) Histogram of # proposed samples

Figure 4.4: Visualization of sample distribution for the non-blind examples in Fig. 4.3. From left to right, the columns show the example with kernel 1,2,3,4. (a) shows the residual between initial intrinsic image and ground truth. (b) shows the map of *accepted* sample energy ed_x. The values of the residuals and sample energy in (a)-(b) are $10\times$ magnified before coded in CMYK colorspace, where *magenta* channel indicates *positive* values and *yellow* negative. (c) shows the normalized distribution of proposed sample locations $\mathbf{p}_i$ (including both *accepted* and *rejected* samples.), coded in *key* channel in CMYK colorspace. (d) shows the histogram of the number of proposed samples in (c). The horizontal axis indicates the bins of the number of proposed samples (clamped at 45 for limited space), and vertical axis indicates the number of pixels at each bin.

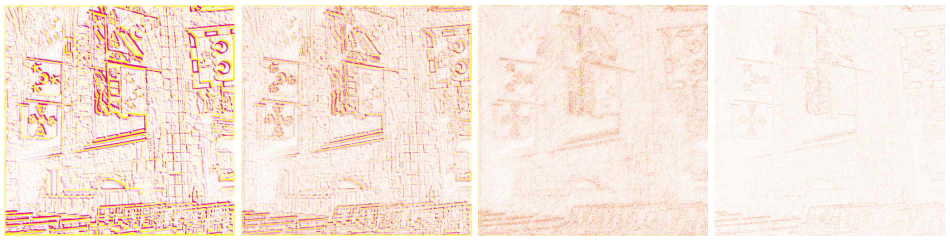

(a) Initial **(b)** Iteration 20 **(c)** Iteration 60 **(d)** Iteration 200

Figure 4.5: Visualization of the residual between the ground truth and our estimated intrinsic image at different iterations, for the example shown in Fig.4.3(a). The values of the residual are $10\times$ magnified before coded in CMYK colorspace, where *magenta* channel indicates *positive* values and *yellow* negative.

finite probability of choosing each possible coordinate axis. This is ensured by the ergodicity of our mutation strategy. For general, non-smooth objectives no such proof exists (see details in [36]), but in Fig. 4.3, we show empirical convergence experiments for our method in the case of non-smooth and non-convex objectives in non-blind intrinsic image estimations.

In Fig. 4.5, we visualize the residual between the true intrinsic and our estimation in selected iterations for the example in Fig. 4.3(a). The algorithm reduces the residual progressively. In Fig. 4.4, we visualize the sample distribution in the random walk process for the examples in Fig. 4.3. As shown in Fig. 4.4(a), the residual is large mostly at pixels near the image edges. Fig. 4.4(b) shows the distribution of *accepted* sample energy ed_x, which are mostly located at pixels where the residual is large. Note that the positive and negative ed_x partially overlap during the random walk process. Fig. 4.4(c) shows the distribution of the *number* of proposed samples (including both *accepted* and *rejected* ones). More samples are proposed near the image edges. Fig. 4.4(d) shows the histogram of the number of proposed samples in Fig. 4.4(c). The majority of pixels consume 5-30 samples.

The intuitive reasons why such an apparently small amount of samples are required are: 1) the samples are not uniform-randomly proposed. The algorithm uses random walk to exploit the spatial coherence in the images, thus enforce importance sampling; 2) the sample energy ed is initialized as a large value to reduce the number of required samples at the beginning (as larger ed can reduce the objective more effectively), and then progressively reduced to better recover smaller details in the image. This multi-weight strategy help reduce the total number of required samples; 3) each proposed sample is evaluated with both positive and negative energy.

4.2.2 Updating the kernel

The updateKernel(.) function is used to perform kernel estimation, i.e., to find the kernel that best explains the blurry input $\mathbf{b}^s$ and intrinsic image $\mathbf{x}^s$ for a given scale s. This operation is performed in derivative domain, minimizing the objective function in Eq. 4.13 consisting of a data term and set of priors $g(.)$ with weights $\theta_{\mathbf{k}}$:

$$f(\mathbf{k}^s) = ||\nabla_{h,v}\mathbf{b}^s - \mathbf{k}^s \otimes \nabla_{h,v}\mathbf{x}^s||_2^2 + \theta_{\mathbf{k}} \cdot g(\mathbf{k}^s), \tag{4.13}$$

where $\nabla_{h,v}$ represents the set of horizontal and vertical first-order derivative operators, i.e., $\nabla_{h,v} = \{\nabla_h, \nabla_v\}$, and thus $\nabla_{h,v}\mathbf{x}^s = \{\nabla_h\mathbf{x}^s, \nabla_v\mathbf{x}^s\}$. We observed that the optimization converges faster when the data-fitting error is computed in the derivative domain rather than in the intensity domain. This is consistent with the findings of Cho *et al* [20]. On the other hand, the cross-channel prior (see Section 4.3.1) requires the image to be represented in the intensity domain. Since mixing the intensity domain and the gradient domain in a single subproblem would be too costly, we use the gradient domain only for the kernel subproblem.

The same stochastic random walk algorithm (Algorithm 5) is used for updating the kernel except samples are now drawn from the kernel image. As before, an energy quantum ed_k is added or removed at each sample location $\mathbf{p}_i$ causing the kernel to be updated by Eq. 4.14, where $\mathbf{k}_i^s$ is the estimated kernel at i^{th} iteration of the random walk, and $\delta_{\mathbf{p}_i}$ is the characteristic function (i.e., Kronecker delta function) for the pixel located at $\mathbf{p}_i$. Non-negativity of the kernel is enforced by rejecting any sample that causes a kernel pixel to become negative.

$$\mathbf{k}_i^s = \mathbf{k}_{i-1}^s \pm ed_k \cdot \delta_{\mathbf{p}_i} \tag{4.14}$$

Updates to the kernel result in a change to the entire blurry image. To evaluate the data efficiently, the algorithm maintains an estimate of the gradient of the current blurry image, $\nabla_{h,v}\hat{\mathbf{b}}_i^s$, which is compared to the down-sampled captured blurry image to evaluate the change to the objective function. Each sample at $\mathbf{p}_i$ is a scaled Dirac function $\pm ed_k \cdot \delta_{\mathbf{p}_i}$, resulting in an update rule whereby a shifted and scaled copy of the current estimate for the intrinsic image $\mathbf{x}^s$ is added:

$$\nabla_{h,v}\hat{\mathbf{b}}_i^s = \nabla_{h,v}\hat{\mathbf{b}}_{i-1}^s \pm ed_k(\delta_{\mathbf{p}_i} \otimes \nabla_{h,v}\mathbf{x}^s) \tag{4.15}$$

As in the intrinsic image update, it is necessary to apply a regularizer to the kernel estimation in order to enforce specific properties. For motion blur kernels we expect kernels to be i) smooth ii) sparse, in the sense that most kernel entries will be zero and iii) continuous, in that the kernel should be a smooth curve over the exposure time. We enforce these properties with the priors from Eq. 4.16-4.18 respectively:

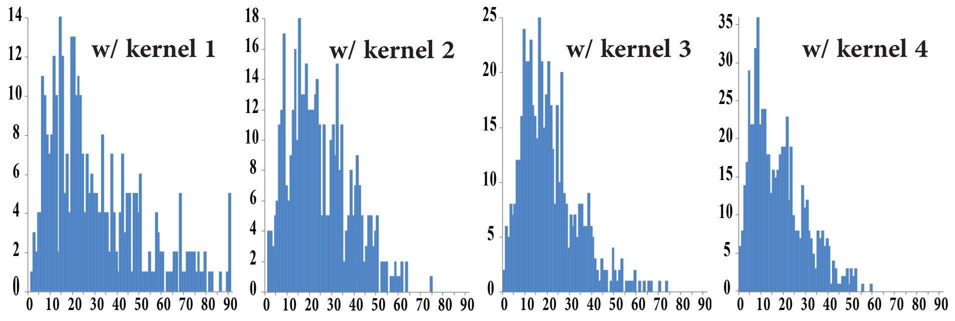
- Smoothness [62]:
$$||\nabla^2 \mathbf{k}^s||_2^2 \tag{4.16}$$

- Sparsity:
$$||\mathbf{k}^s||_p, 0 \leq p < 1 \tag{4.17}$$

- Continuity:
$$||\mathbf{k}^s - AD(\mathbf{k}^s)||_2^2 \tag{4.18}$$

For the continuity prior, Eq. 4.18, anisotropic diffusion [81] is used for the filter AD(.). This is a non-linear filter that favors long continuous features which helps to reconstruct thin motion-blur trails. A benefit of the stochastic optimization algorithm is that this function may be computed exactly for each sample, rather than linearized per iteration. As with the intrinsic update, the key benefit of the stochastic framework is that only local evaluations of the regularizers are required.

After updating the kernel, a simple denoising filter is applied that sets any pixel in $\mathbf{k}^s$ to zero whenever its eight neighboring pixels are near zero. This removes isolated speckles that sometimes occur with the stochastic random walk. The pixels whose intensity is lower than a threshold (i.e., 0.05 times the highest pixel intensity of current estimated kernel) are also set to be zero. This can be interpreted as an



(**a**) Histogram of # proposed samples

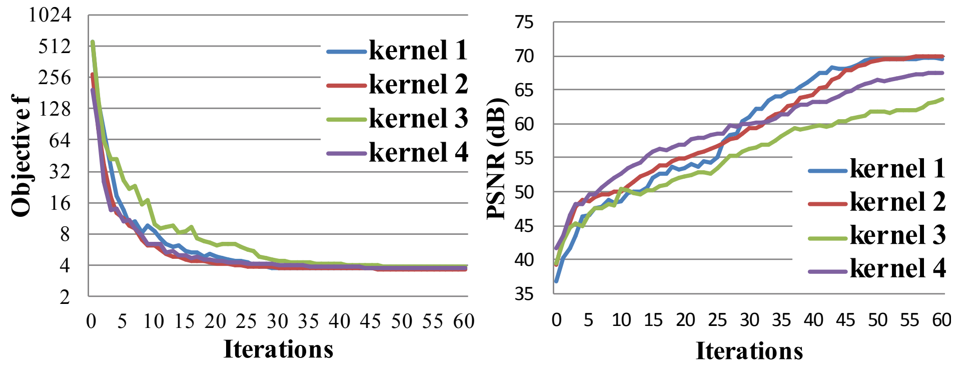


Figure 4.6: Example of *non-blind* kernel estimation. The true intrinsic image and kernel are given in Fig. 4.2. The input blurry images are given in Fig. 4.3. (a) shows the histogram of the number of proposed samples (including both *accepted* and *rejected* samples). The bottom row shows the change of objective function f and PSNR as the number of iterations increases. The initial kernels can be arbitrary for non-blind kernel estimation. Note that the sample energy is non-zero at the region where the pixel intensity is zero in both initial and final recovered kernel. This is due to the post-processing (remove isolated pixel, shrinkage, and normalization) at the end of each iteration. This post-processing also causes non-monotonicity in the objective and PSNR curve. The priors defined in Eq. 4.16, 4.17 and 4.18 were applied. In each iteration, 200 samples were proposed.

additional shrinkage operator that ensures kernel sparsity. The kernel image is normalized to 1 at the end of each iteration.

In Fig. 4.6, we show empirical convergence test of our method for non-blind kernel estimations. Regarding the parameters in Algorithm 5, we use the initial value of ed_k as 0.01, and γ as 0.1 in the experiments.

4.2.3 Updating the weights

The weights $\theta_{\mathbf{x}}$ and $\theta_{\mathbf{k}}$ define the relative strength of the data-fitting error and regularizers in the objective functions for intrinsic image and kernel updates. The algorithm begins with initially high $\theta_{\mathbf{x}}$ (except for the cross-channel prior explained in Section 4.3.1) at the coarsest scale and halves them whenever a new scale is started, until minimum thresholds are reached. This helps to avoid local optima in the subproblem. $\theta_{\mathbf{k}}$ is kept unchanged for all scales in our experiments.

After the kernel is estimated at the finest scale, the function updateIntrinsicImage(.) is applied again to generate the final estimation of intrinsic image $\mathbf{x}$ (i.e., line 18, Algorithm 4).

In Fig. 4.7, we visualize the progress of intrinsic and kernel estimation at multi-scale process.

4.3 Algorithmic Extensions

Having described the basic algorithm in Section 4.2 we now proceed to introduce several useful extensions, including non-convex priors for color images (Section 4.3.1) and chromatic kernels (Section 4.3.2), as well as partially saturated pixels (Section 4.3.3). Finally, we demonstrate how non-linear versions of the image formation model can also be included to account for non-Gaussian noise models (Section 4.3.4).

4.3.1 Color images

To recover color images corrupted by motion blur, a simple extension of the basic algorithm might perform kernel estimation as described in Section 4.2.2 (summing the data term over all three channels) followed by separately deblurring each intrinsic channel. However, better results can be obtained by jointly deblurring all

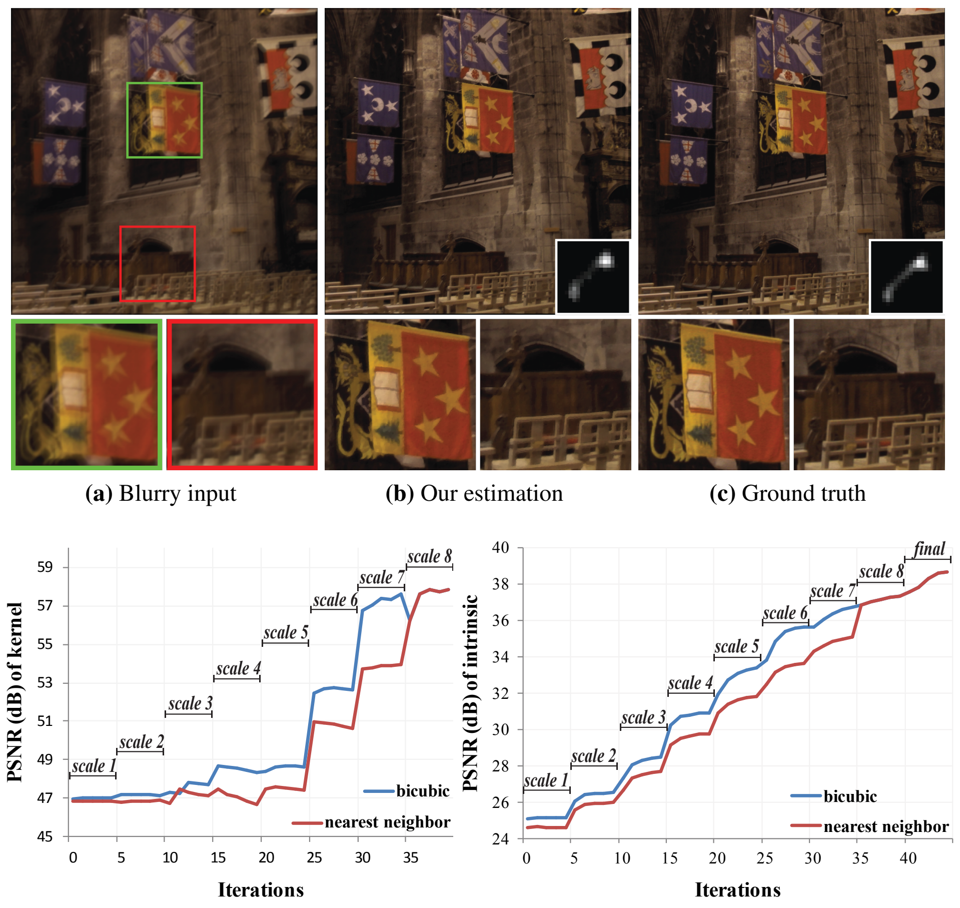
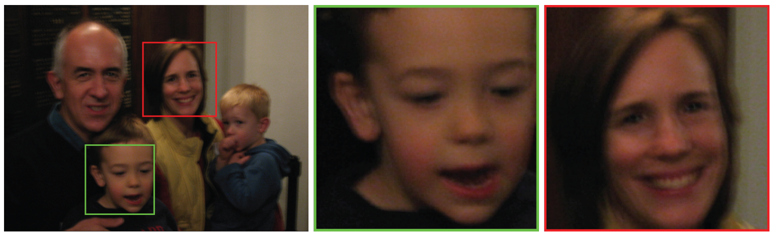
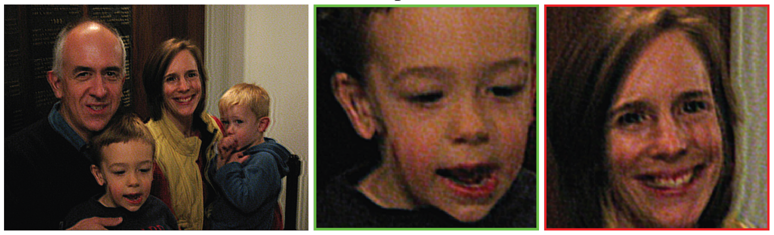


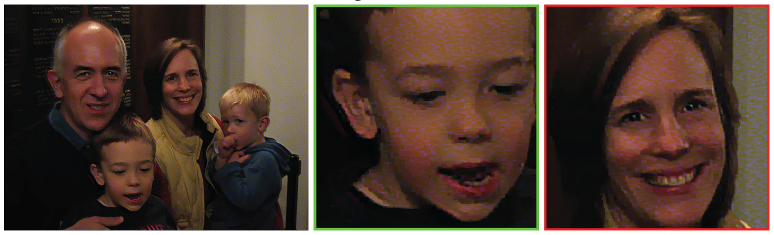
Figure 4.7: Example of our blind estimation of intrinsic and kernel. The two plots show the PSNR value of intermediate estimation of intrinsic and kernel at multi-scale scheme. To compute the PSNR values, at each scale we upsample the intrinsic and kernel to the finest resolution by bicubic or nearest neighbor and remove the possible shifts first. The "final" step in the plot means the final restoration of the intrinsic image, i.e., line 18, Algorithm 4.



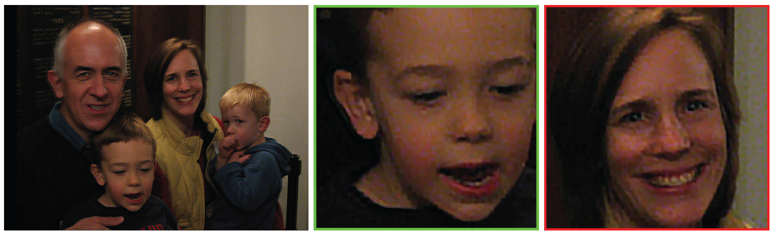
(a) Input



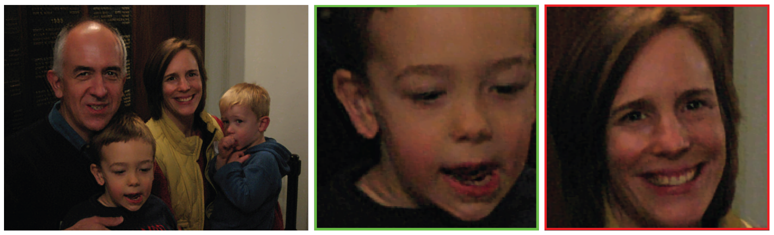
(b) Fergus *et al* [27]



(c) Cho *et al* [20]

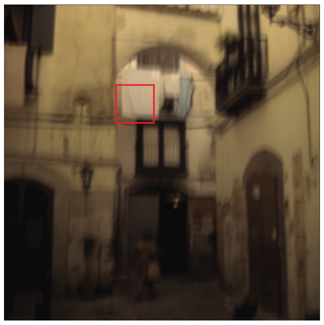


(d) Xu *et al* [1]

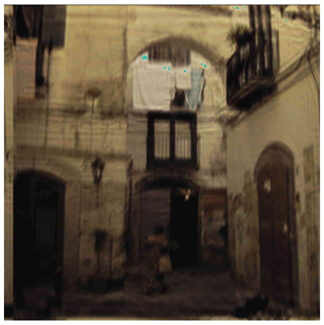


(e) Ours

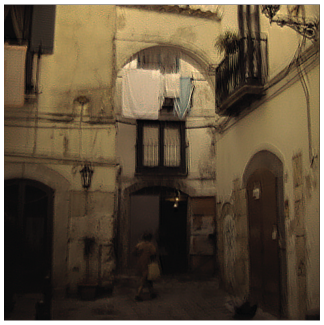
Figure 4.8: Results on a noisy real-world image. Our algorithm significantly reduces chromatic artifacts compared with previous methods (better view on screen).



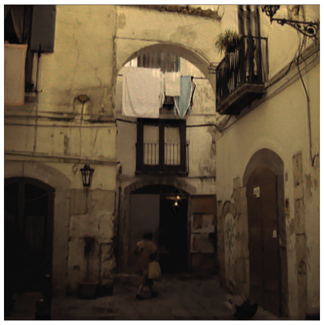
(a) Input



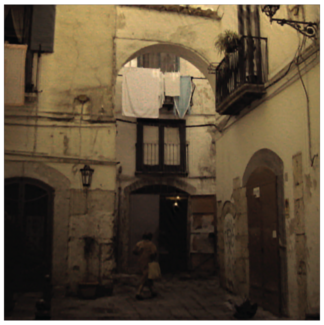
(b) Fergus *et al* [27]



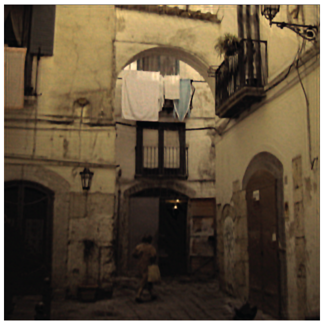
(c) Shan *et al* [98]



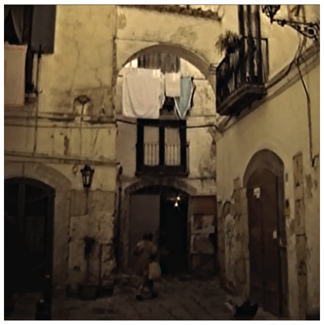
(d) Cho *et al* [20]



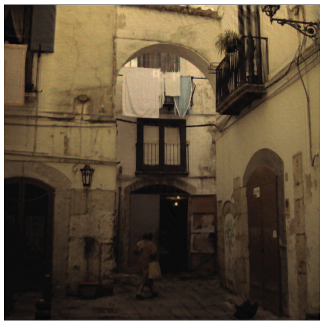
(e) Xu & Jia [113]



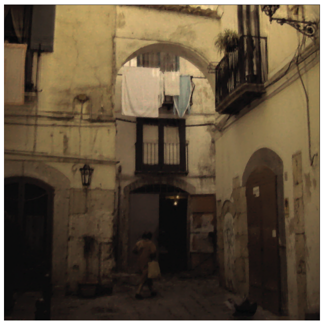
(f) Hirsch *et al* [46]



(g) Krishnan *et al* [59]

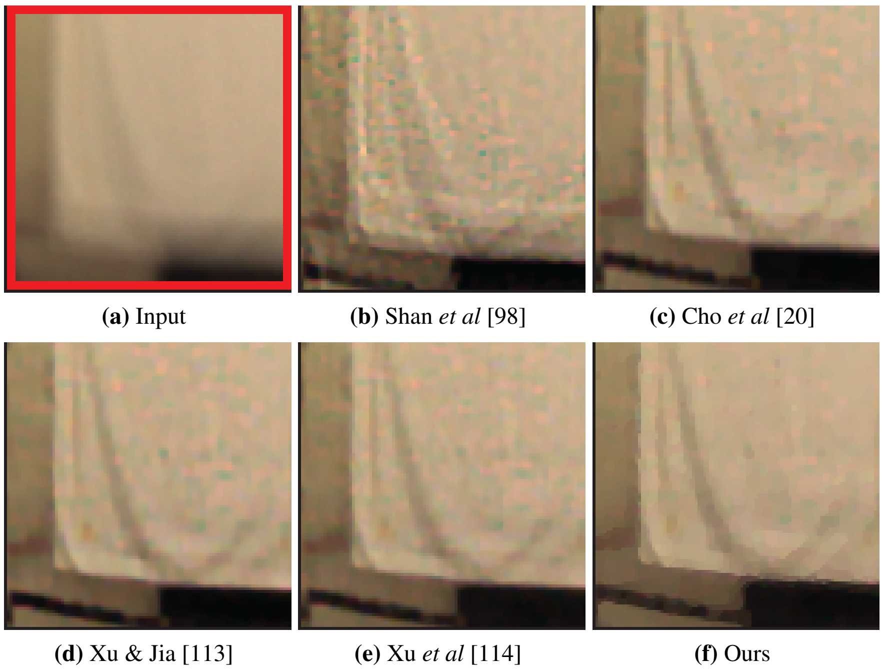


(h) Xu *et al* [114]



(i) Ours

Figure 4.9: Results on a real-world image and visual comparisons of state-of-the-art methods. A closeup is shown in Fig. 4.10.


(a) Input **(b)** Shan *et al* [98] **(c)** Cho *et al* [20]

(d) Xu & Jia [113] **(e)** Xu *et al* [114] **(f)** Ours

Figure 4.10: Closeup of the cloth region in the scene shown in Fig. 4.9. Our result contains much fewer chromatic artifacts than the other methods.

intrinsic channels simultaneously since the majority of edges in the true intrinsic image occur in all channels, with sparse hue changes. Based on this observation, Heide *et al* [43] proposed a cross-channel prior to remove chromatic aberrations caused by low-quality lenses:

$$\sum_{i,j\in\{r,g,b\}} \lambda_{ij}||\mathbf{x}_i^s \cdot \nabla_{h,v}\mathbf{x}_j^s - \mathbf{x}_j^s \cdot \nabla_{h,v}\mathbf{x}_i^s||_p, 0 < p \leq 1 \tag{4.19}$$

Adding the cross-channel priors to the regularizers $g(.)$ for the intrinsic image results in a non-convex objective. Heide *et al*used an alternating minimization in which one channel is deblurred with the other two fixed. We instead reconstruct all channels simultaneously by running one sampling chain per channel run in lock-step. Although the method still alternates between the channels, this occurs so

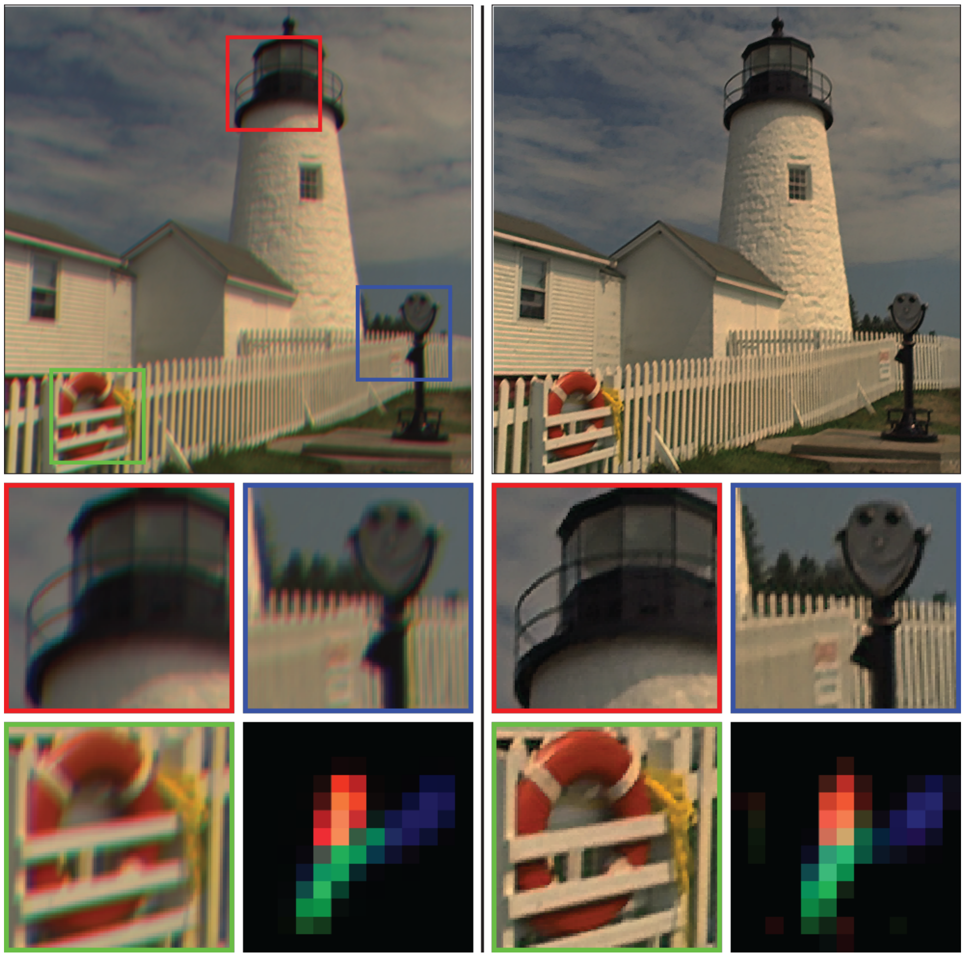


Figure 4.11: Results on chromatic blur kernel. *Left column*: input image blurred with a chromatic kernel (right bottom); *Right column*: our recovered intrinsic image and blur kernel (right bottom).

frequently that the optimization is effectively performed simultaneously over all channels. Our algorithm begins with low weight for cross-channel prior at the coarsest scale and doubles it when a new scale is started.

We find that the cross-channel prior proposed in [43] improves deblurring performance even for achromatic kernels by suppressing color artifacts that would be introduced by separate deblurring of each channel. Example comparisons are shown in Fig. 4.8, 4.9 and 4.10.

4.3.2 Chromatic kernels

It is further possible to extend the method to estimating chromatic kernels that occur in sensor fusion where individual channels have unique exposure times (e.g. [60]). This is accomplished by separately updating each kernel channel, but using the cross-channel prior in Eq. 4.19 during the intrinsic image update at each scale as described in Section 4.3.1. Synthetic examples of this strategy are shown in Fig. 4.11.

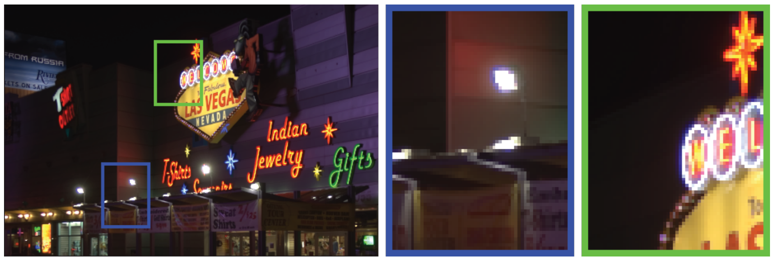
4.3.3 Saturated or missing data

Saturated pixels are a common occurrence when taking photos with consumer cameras, as is missing or unreliable data due to lens debris. Deblurring data with saturated pixels often results in visually objectionable ringing artifacts since the capture process clamps the input data in a way that is not consistent with the image formation model, while for debris it may be preferable to mask out such regions and allow the deblurring algorithm to inpaint plausible content. In the following discussion, we consider only the case of saturated blurry pixels, however the approach applies equally well to lens debris.

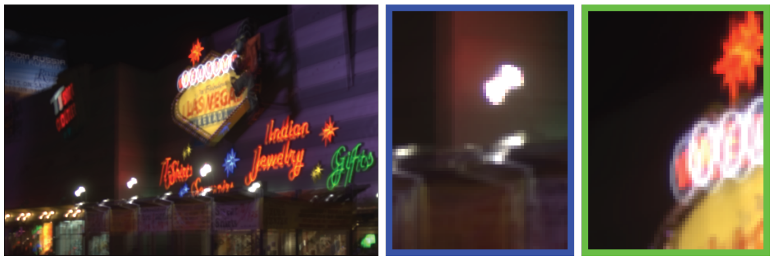
To handle such saturated pixels, our algorithm performs kernel estimation as usual using all non-saturated pixels. When reconstructing the final intrinsic image, previous work, including [36], simply uses a data term that omits saturated blurry image pixels, leading to improved results over deblurring naively. However we have found that a two-phase approach to intrinsic image estimation yields much improved results.

The two phase algorithm divides the intrinsic image into two regions: a *reliable* region which does not contribute to saturated blurry image pixels and an *unreliable* region that contains pixels do contribute saturated blurry pixels. Four binary masks are defined:

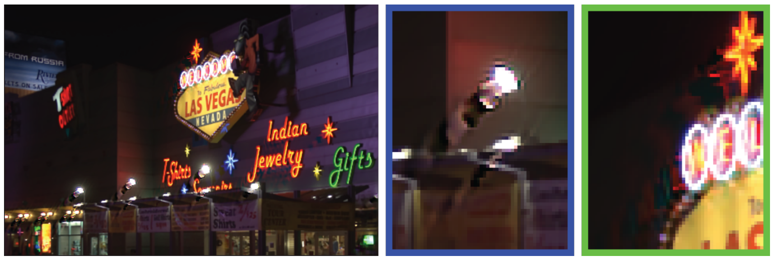
- $\mathbf{m}_s^{\mathbf{b}}$ Saturated pixels in the blurred input.
- $\mathbf{m}_v^{\mathbf{x}}$ Mask of unreliable intrinsic pixels with a saturated pixel from $\mathbf{m}_s^{\mathbf{b}}$ in their support.
- $\mathbf{m}_u^{\mathbf{x}}$ Mask of reliable intrinsic pixels, the inverse mask of $\mathbf{m}_v^{\mathbf{x}}$.



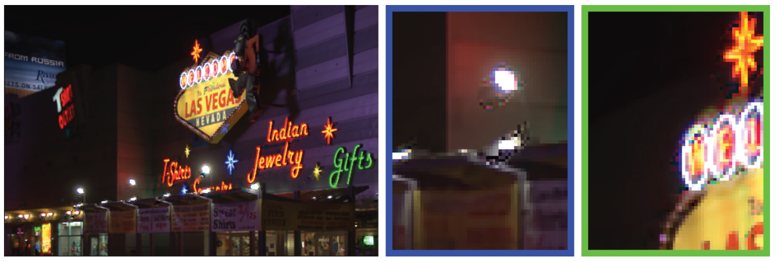
(a) True intrinsic image



(b) Input blurry image



(c) Recovered intrinsic *without* two-phase approach



(d) Recovered intrinsic *with* two-phase approach

Figure 4.12: Results on partially saturated image. The dynamic range of pixel intensities in the true intrinsic image is [0, 68]. The simulated blurry image is clamped to 1. The proposed two-phase approach helps reduce ringing artifacts near saturated pixels.

- $\mathbf{m}_d^{\mathbf{b}}$ Mask of blurred pixels with a contribution from an unreliable pixel, i.e., where $\mathbf{k} \otimes \mathbf{m}_v^{\mathbf{x}} \neq 0$.

Using these masks we perform the intrinsic image reconstruction in two phases. First the intrinsic image is estimated for reliable intrinsic image pixels in $\mathbf{m}_u^{\mathbf{x}}$, masking out data term contributions from unreliable blurred pixels in $\mathbf{m}_d^{\mathbf{b}}$ by minimizing:

$$f(\mathbf{x}) = ||(\mathbf{b} - \mathbf{k} \otimes \mathbf{x}) \cdot (\mathbf{1} - \mathbf{m}_d^{\mathbf{b}})||_2^2 + \theta_{\mathbf{x}} \cdot g(\mathbf{x}) \tag{4.20}$$

This optimization outputs the estimated intrinsic image everywhere that the linear image formation model holds, generating samples everywhere in the image as needed to minimize both data term *and* the priors $g(.)$. The second phase reconstructs the unreliable regions, leaving the intrinsic image in the reliable regions (i.e. in $\mathbf{m}_u^{\mathbf{x}}$) fixed.

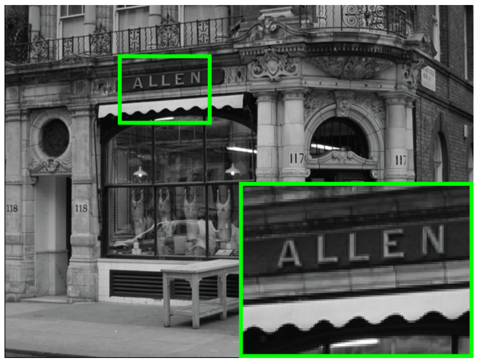
$$f(\mathbf{x}) = ||(\mathbf{b} - clip(\mathbf{k} \otimes \mathbf{x})) \cdot \mathbf{m}_d^{\mathbf{b}}||_2^2 + \theta_{\mathbf{x}} \cdot g(\mathbf{x}) \tag{4.21}$$

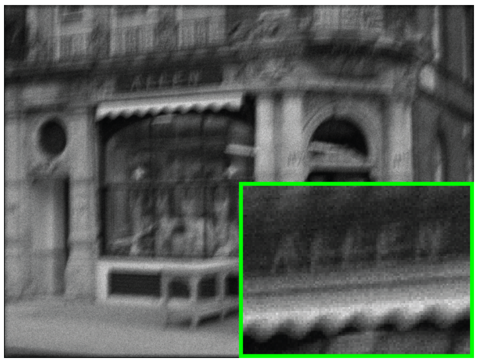
The second phase only generates samples within the mask $\mathbf{m}_v^{\mathbf{x}}$. By performing the reconstruction in this method, ringing is constrained to the non-reliable image region unlike in the typical approach where it can spread well beyond as a consequence of the data fitting term. Fig. 4.12 shows our results on synthetic partially saturated data. When dealing with color images, the proposed two-phase reconstruction is the same as described except that the masks vary in different color channels.

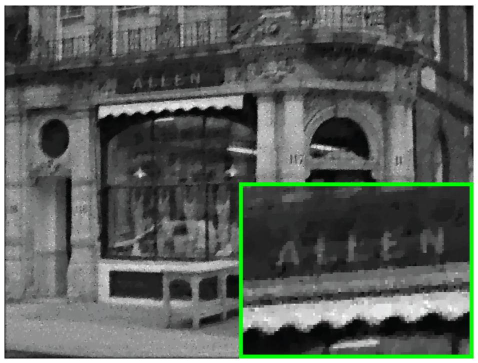
4.3.4 Poisson noise

In previous sections, we use quadratic fidelity in the objective by assuming white Gaussian noise in the input images. Here we extend our algorithm to deal with images containing Poisson noise, using the Anscombe transform [4]:

$$Ansc(\mathbf{z}) = 2\sqrt{c \cdot \mathbf{z} + 3/8}, \tag{4.22}$$


(a) True intrinsic


(b) Input blurry (21.23 dB)


(c) Ours, Gaussian (24.24 dB)

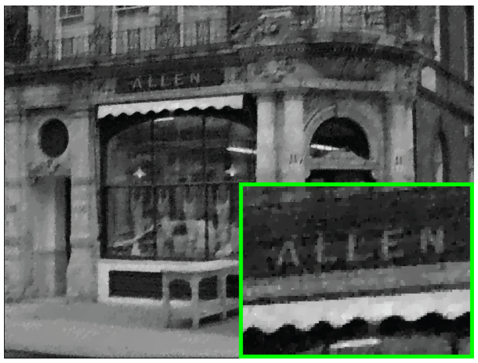

(d) Ours, Poisson (24.70 dB)

Figure 4.13: Results on a synthetic image with Poisson noise (peak intensity = 500). Each subfigure contains an inset at the right-bottom corner for better view. (c) shows our result with Gaussian noise assumption (using Eq. 4.2, 4.13). (d) shows our result with Poisson noise assumption and the Anscombe transform (using Eq. 4.23, 4.24). The result with Poisson noise assumption recovers more details and is less noisy especially in bright regions. We run both experiments with numerous parameters and select the results with highest PSNR.

where vector $\mathbf{z}$ represents the input image, c is a scalar for converting $\mathbf{z}$ to its corresponding photon number.

The Anscombe transform (denoted as Ansc(.)) converts Poisson noise to approximately Gaussian noise. It allows us to use quadratic term for data-fitting error in the transform domain, and thus fits our multi-scale framework. Similar transformations are available for other noise models, including mixtures of Gaussian and Poisson noise, which are common in real images.

Specifically, as shown in Eq. 4.23 and 4.24, we apply the Anscombe transform on the observed blurry image $\mathbf{b}$ before downsampling it into each scale s. At each scale, the data-fitting error is computed in the transform domain, while the regularizers on the intrinsic image $g(\mathbf{x}^s)$ are still computed in regular domain. This is because all the intrinsic image priors were learned from natural images, and may not hold well in the transform domain.

$$f(\mathbf{x}^s) = ||(Ansc(\mathbf{b}))^s - Ansc(\mathbf{k}^s \otimes \mathbf{x}^s))||_2^2 + \theta_{\mathbf{x}} \cdot g(\mathbf{x}^s) \tag{4.23}$$

$$f(\mathbf{k}^s) = ||\nabla_{h,v}(Ansc(\mathbf{b}))^s - \nabla_{h,v} Ansc(\mathbf{k}^s \otimes \mathbf{x}^s)||_2^2 + \theta_{\mathbf{k}} \cdot g(\mathbf{k}^s) \tag{4.24}$$

Note that the data-fitting terms are non-convex now. In Fig. 4.13, we show a synthetic example with Poisson noise (peak intensity = 500). We run our framework with Gaussian noise assumption (using Eq. 4.2, 4.13), and with Poisson noise assumption (using Eq. 4.23, 4.24). The latter produces visually and quantitatively better results.

4.4 Results

Visual comparisons

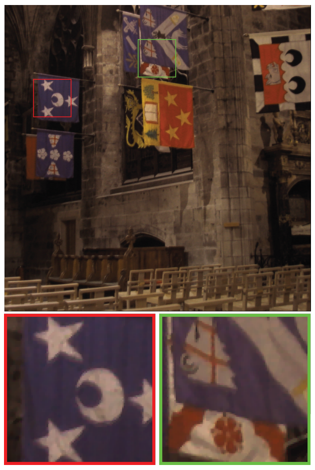
In Fig. 4.8, 4.9, and 4.10 we compare the results of our algorithm with several state-of-the-art methods on real-world images. Our algorithm significantly reduces chromatic artifacts in the recovered intrinsic images. Fig. 4.11 shows our results on simulated data with chromatic blur. Our algorithm recovers the chromatic kernel well and produces a clean intrinsic image without color artifacts. Fig. 4.12 contains results for partially saturated data. The proposed simple two-phase method effectively reduces the ringing artifacts near the saturated pixels. Fig. 4.13 shows our results with the proposed non-convex data-fitting error in Eq. 4.23 and 4.24.

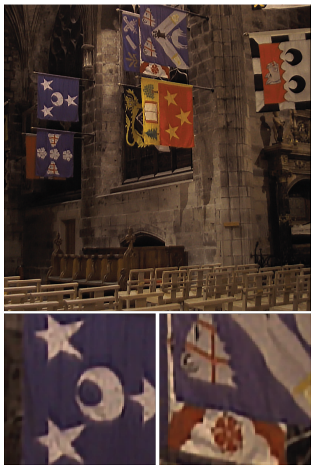
Qualitative comparisons

We also tested our algorithm on a real-world database that contains both ground truth intrinsic images and kernels [57]. The database consists of 4 images, each of which is blurred with 12 kernels. 8 of the kernels are approximately uniform across the image, while 4 kernels are both large and exhibit strong spatial variation. The dataset provides 199 unblurred frames recorded during the camera motion trajectory for each image. We run the provided script to compute the PSNR value for each of our results. The script first estimate the optimal intensity scaling and translation between the recovered image and the unblurred reference image such that their ℓ_2 error over three color channels becomes minimal. PSNR is then computed using these calibrated images. The final PSNR values reported in the paper is defined as the maximum PSNR between the recovered image and any of the 199 unblurred reference images along the trajectory.

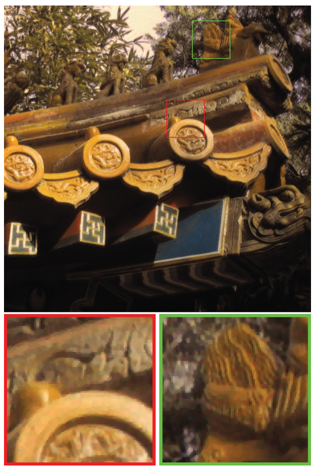
In Table 4.1, we show the PSNR values averaged over all 4 images for each kernel by our algorithm, and compare with Fergus *et al* [27], Shan *et al* [98], Cho *et al* [20], Xu and Jia [113], Krishnan *et al* [59], Whyte *et al* [106], Hirsch *et al* [46] and Xu *et al* [1]. The downloadable software by Xu *et al* [1] incorporates technologies proposed in [113] and [114]. We adjusted the parameters in their software to produce the best results possible. For the other methods we use the published PSNR results directly from the database [57]. The PSNR value and recovered intrinsic image for each image can be found in the supplementary material.

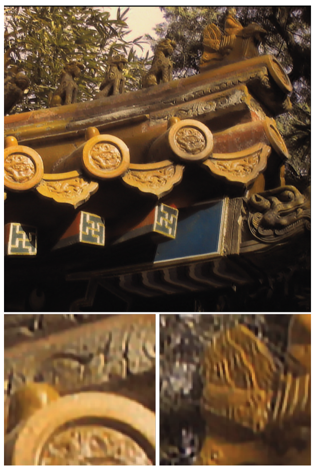
On mostly spatially-invariant kernels (#1-7 and #12), our algorithm produces results that are either close to or better than the best published methods. We notice that the state-of-the-art Xu *et al* [1]'s results sometimes look sharper than ours, but are actually oversharpened at the edges and thus have lower PSNRs. We show examples in Fig. 4.14, where their results even look sharper than the ground truth and contain halo artifacts at the edge pixels. This may be caused by the use of explicit shock filter and bilateral filter in their kernel estimation.


(a) Ours (38.82dB)


(b) Xu [1] (33.34dB)

(c) Reference


(d) Ours (33.29dB)


(e) Xu [1] (30.72dB)

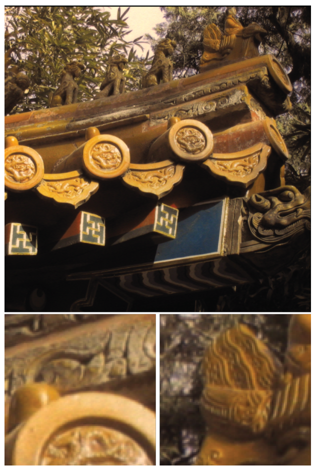

(f) Reference

Figure 4.14: Comparison on image #1 with kernel #3 and image #4 with kernel #4 from the dataset [57]. The filter-based method Xu *et al* [1] over-sharpens the image and creates halo artifacts near the edges (better view on screen).

Table 4.1: PSNR (dB) comparisons on the benchmark dataset [57]. On mostly spatially-invariant kernels (#1-7 and #12), our algorithm produces results close to or better than the state-of-the-art methods. Our algorithm fails to produce the best results on the extremely large and spatially-variant kernels #8-11 (with size over 141 by 141 pixels). The resolution of each input image is 800 by 800 pixels. Please see Section 4.4 for more details.

	Kernel 01	Kernel 02	Kernel 03	Kernel 04	Kernel 05	Kernel 06	Kernel 07	Kernel 12
Kernel width	35	35	17	35	35	35	35	49
Input	24.89	27.85	32.34	28.09	29.22	25.22	24.94	23.85
Fergus [27]	20.63	29.38	31.51	29.48	25.76	22.34	20.70	20.11
Shan [98]	29.42	28.28	30.77	28.60	30.04	26.85	26.83	23.76
Cho [20]	32.49	31.86	31.44	30.89	32.38	30.01	30.91	28.98
Xu and Jia [113]	32.45	32.58	32.22	32.01	**32.98**	30.43	**31.40**	29.51
Krishnan [59]	31.57	31.09	31.67	30.77	30.58	24.59	25.80	23.32
Whyte [106]	32.08	32.20	34.61	32.10	32.54	30.89	29.06	28.21
Hirsch [46]	32.04	30.09	33.94	32.13	32.82	29.53	29.32	26.81
Xu *et al* [1]	31.94	31.71	31.42	31.30	31.78	30.06	30.87	29.18
Ours	**32.65**	**32.68**	**35.35**	**33.74**	32.56	**31.74**	30.96	**30.12**

	Kernel 08	Kernel 09	Kernel 10	Kernel 11
Kernel width	141	141	141	141
Input	19.55	19.82	20.50	22.90
Fergus [27]	17.93	18.87	18.72	16.95
Shan [98]	19.19	22.90	20.49	23.56
Cho [20]	22.34	28.00	23.96	24.54
Xu and Jia [113]	**22.54**	**28.35**	**24.12**	**25.87**
Krishnan [59]	15.35	22.14	20.15	21.68
Whyte [106]	19.07	19.80	20.38	23.19
Hirsch [46]	19.49	23.50	20.19	23.40
Xu *et al* [1]	19.52	26.46	21.77	25.77
Ours	19.79	21.16	21.02	22.88

All methods perform significantly worse on the spatially-variant kernels #8-11. Our software does not currently deal with spatially-variant kernels, and instead recovers an average kernel for the whole image. As a result, our method only produces PSNR values in the middle of the field for these kernels. Fixing this problem would require cutting the image into tiles, solving a blind deconvolution problem for each tile, realigning the resulting tiles (since blind deconvolution can introduce an offset in the kernel and intrinsic image), and stitching the results back together. This should be easily feasible in the future, but is not currently implemented.

Computational cost

We compared the runtime with two state-of-the-art methods, Cho *et al* [20] and Xu *et al* [1], using the executable files provided by the authors. The method by Cho *et al* [20] requires only a few seconds per megapixel, but their results are consistently worse than ours (see Fig. 4.8, 4.9 and 4.10 and Table 4.1). In Fig. 4.8 for an 848 by 636 blurry/noisy RGB image and a 19 by 19 achromatic kernel, our method requires 197.0 seconds in total (140.4 seconds for blind kernel estimation, and 56.6 seconds for non-blind deconvolution). Xu *et al*'s method [1] is relatively faster than ours at 121.9 seconds, but their results show suffer from more artifacts.

We also run our algorithm on different size images and kernels and report the runtime in Table 4.2. The code was compiled with gcc and the experiments were done on an Intel i7 CPU with 16GB RAM. The result images and parameters are shown in the supplementary document.

Influence of the noise

To test the influence of noise on the performance, we run our algorithm on synthetic data with various noise levels. The parameters are tuned roughly for the results. The results are shown in Fig. 4.15.

Priors and parameter selection

Our framework allows us to easily adapt any priors or data-fitting term in the objective function. We use *smoothness* (Eq. 4.16), *sparsity* (Eq. 4.17, $p = 0.8$) and *continuity* (Eq. 4.18) as kernel priors for all results in the paper. We use *sparse*

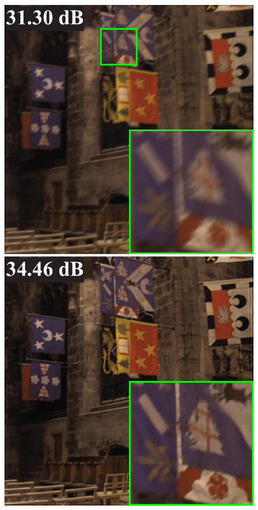
Table 4.2: Run-time analysis. The reported time are in seconds. We run our unoptimized code on images with pixel resolution 400×400, 800×800, 1200×1200, and kernels with pixel resolution 15×15, 25×25, 35×35. The blurry images are simulated with resized intrinsic image and blur kernel at each resolution. (a) shows the experiments on gray-scaled images. (b) shows the experiments on RGB images, where all channels were recovered simultaneously. We increased the number of proposed samples as the image size and kernel size increase. The result images and parameters are shown in the supplementary.

(a) Experiments on gray-scaled images

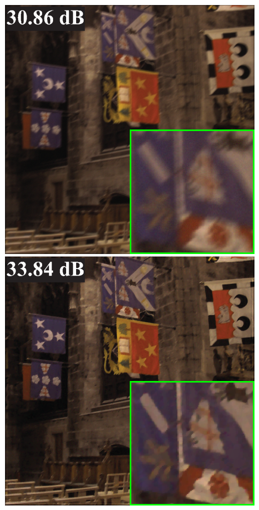
Image width	Kernel width	Multiscale estimation		Final restoration	Total
		Intrinsic update	Kernel update		
400	15	4.68	7.61	2.92	15.71
	25	13.44	14.58	5.45	34.03
	35	23.24	21.44	15.74	61.12
800	15	11.53	54.64	7.05	74.93
	25	19.94	59.64	14.24	95.48
	35	53.81	80.31	37.03	172.99
1200	15	18.15	132.56	13.06	167.85
	25	30.63	139.03	25.81	199.09
	35	75.55	186.19	53.85	319.42

(b) Experiments on color images

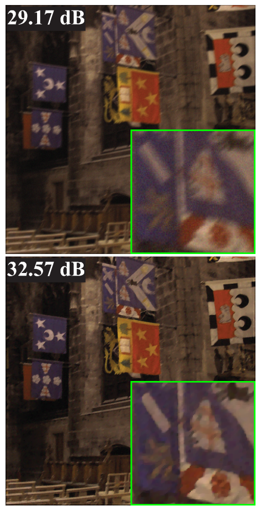
Image width	Kernel width	Multiscale estimation		Final restoration	Total
		Intrinsic update	Kernel update		
400	15	13.02	25.96	8.13	47.62
	25	36.67	49.74	16.24	103.21
	35	65.02	72.77	47.33	185.81
800	15	32.58	180.57	19.94	234.81
	25	56.32	187.81	40.33	286.04
	35	155.47	439.09	116.72	713.16
1200	15	48.99	402.35	35.83	491.09
	25	83.42	408.32	71.85	567.12
	35	217.86	594.26	154.54	971.14



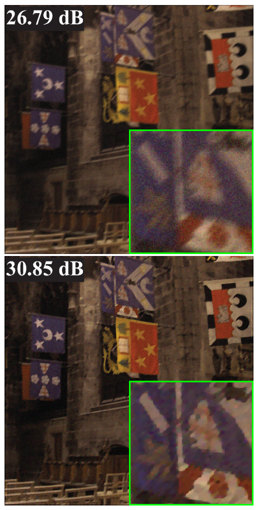
(a) $\sigma = 0$



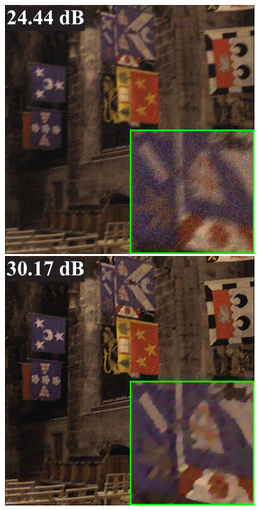
(b) $\sigma = 0.01$



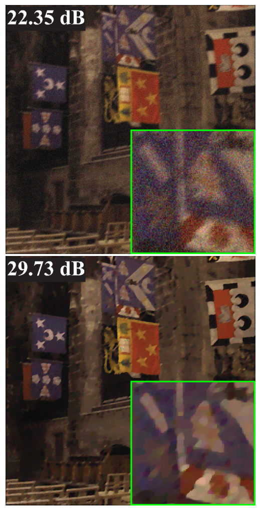
(c) $\sigma = 0.02$



(d) $\sigma = 0.03$



(e) $\sigma = 0.04$



(f) $\sigma = 0.05$

Figure 4.15: Results on synthetic data with difference levels of white Gaussian noise. In subfigure (a-f), the standard derivation (σ) of the noise is set to be 0, 0.01, 0.02, 0.03, 0.04 and 0.05 respectively. In each subfigure, the 1st row shows the input blurry image, and the 2nd row shows our deblurred result. The PSNR values are shown at the left-top corner of each image. An inset is shown at the right-bottom corner of each image for better view.

gradient (Eq. 4.11, p = 0.6 or 0.8), *sparse second-order derivatives* (Eq. 4.12, p = 1) and *cross-channel prior* (Eq. 4.19, p = 1) as intrinsic image priors for Fig. 4.7, 4.8, 4.9, 4.10 and Table 4.1. And we use *isotropic total variation* (Eq. 4.9) and *cross-channel prior* (Eq. 4.19, p = 1) as intrinsic image priors for Fig. 4.11.

Regarding the number of iterations and sample mutations in our experiments, we usually use T (in Algorithm 1) as 5-10, N (in Algorithm 2) as 5 for both intrinsic and kernel updating, and M (in Algorithm 2) as 10000-50000 for intrinsic updating and 100-500 for kernel updating in the multi-scale process. In the final intrinsic image restoration step (line 18, Algorithm 4), we usually use N as 5, and M as 100000-500000. These numbers are proportionally adjusted with the resolution of input image and kernel for better convergence.

The prior weights $\theta_{\mathbf{x}}$ and $\theta_{\mathbf{k}}$ are roughly tuned for best results. In our experiments we set the initial and minimum-threshold weight of *sparse gradient* as 0.05-0.5 and 0.0005-0.002, the initial and maximum-threshold weight of *cross-channel prior* as 0.0001 and 0.0005 (see Section 4.2.3 for the strategy of weights updating). In the final intrinsic image restoration step, we set the weight of *sparse gradient* and *cross-channel prior* as 0.0005-0.002 and 0.0005-0.002.

4.5 Conclusion and future work

In this paper we present an attempt using simple random search technique for complex imaging problems: a simple and effective algorithm for blind motion deblurring from a single input image. We propose to use cross-channel information to reduce chromatic artifacts in the estimated intrinsic images and to recover chromatic blur kernels. We also propose a two-phase method to reduce ringing artifacts when deblurring saturated or missing pixels. Furthermore, we propose to use a non-convex data-fitting term to deal with Poisson noisy images.

Our algorithm provides an easy-to-use framework for blind deconvolution problems. It allows us to easily test new priors for both the kernel and the intrinsic image. This kind of experimentation would be much harder with other optimization methods.

The computational efficiency of our method is below those highly optimized specialized solvers. However, such solvers typically include only a single regular-

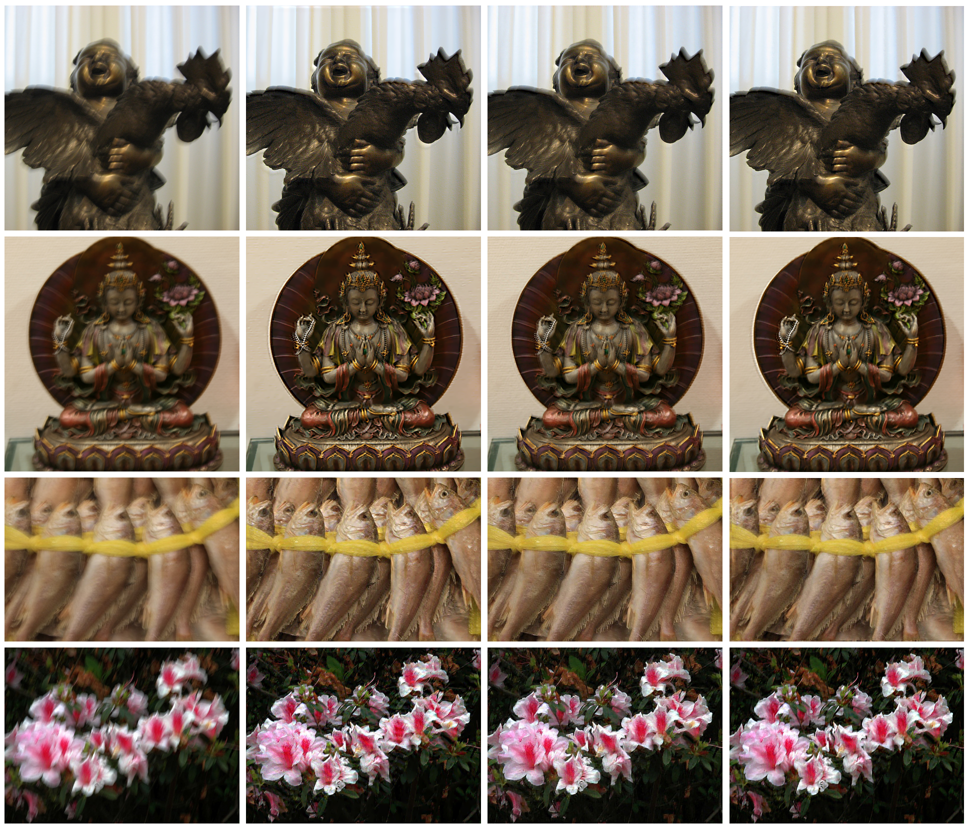

(a) Blurry input **(b)** Cho *et al* [20] **(c)** Xu *et al* [1] **(d)** Ours

Figure 4.16: Results on more real data from [20] and [1].

ization term, whereas we can easily combine many, for improved image quality.

In the future, we would like to extend our algorithm to handle spatially variant kernels with the method outlined above. We also would like to further improve on the handling of saturated pixels. We observe that pixels saturated in one channel might not be saturated in another. By employing the cross-channel information together with neighboring pixels, we should be able to recover the saturated pixels better.

Chapter 5

Document Image Deblurring

5.1 Introduction

In this chapter, we propose a new algorithm for practical document deblurring that achieves both high quality and high efficiency. In contrast to previous works relying on low-order filter statistics, our algorithm aims to capture the domain-specific property of document images by learning a series of scale- and iteration-wise high-order filters. A motivational example is shown in Fig. 5.1, where we compare small patches extracted from a natural image, a large-font text image and a common text document image. Since most deblurring methods adopt a multi-scale framework in order to avoid bad local optima, we compare patches extracted from multiple scales. Evidently, the natural image and large-font text image both contain long, clear edges at all scales, making the use of sparse gradient priors effective. In contrast, patches from the document image with a small font size are mostly composed of small-scale high-order structures, especially at coarse scales, which makes sparse gradient priors to be inaccurate. This observation motivates us to use high-order filter statistics as effective regularization for deblurring document images. We use a discriminative approach and learn such regularization terms by training a multi-scale, interleaved cascade of shrinkage field models [92], which was recently proposed as an effective tool for image restoration.

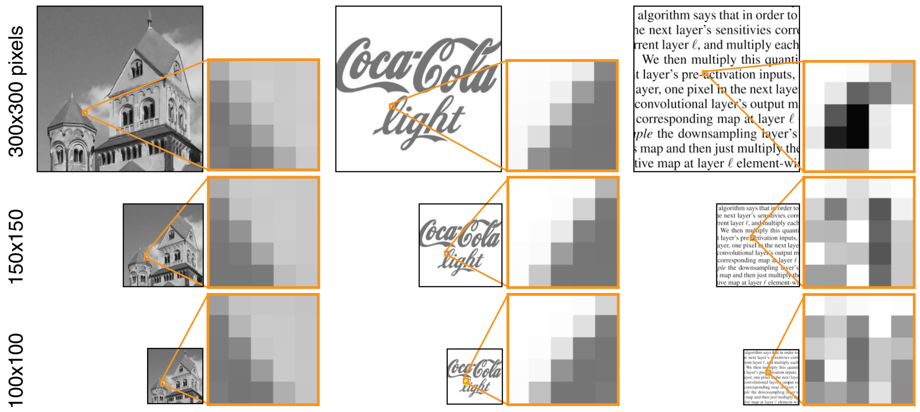


Figure 5.1: Visual comparison between a natural image (left), a large-font text image (middle) and a common text document image at 150 PPI (right) at various scales.

Our main contributions include:

- We demonstrate the importance of using high-order filters in text document image restoration.
- We propose a new algorithm for fast and high-quality deblurring of document photographs, suitable for processing high resolution images captured by modern mobile devices.
- Unlike the recent Convolutional Neural Network (CNN) based document deblurring method [48], our approach is robust to page orientation, font style and text language, even though such variants are not included at our training.

5.2 Method

As in most previous work, we assume a simple image formation model for each local text region as

$$\mathbf{b} = \mathbf{K}\mathbf{x} + \mathbf{n}, \tag{5.1}$$

where $\mathbf{b}$ represents the degraded image, $\mathbf{x}$ the sharp latent image, matrix $\mathbf{K}$ the corresponding 2D convolution with blur kernel $\mathbf{k}$, and $\mathbf{n}$ white Gaussian noise.

The goal of the post-processing is to recover $\mathbf{x}$ and $\mathbf{k}$ from single input $\mathbf{b}$.

The Shrinkage Field (SF) model has been recently proposed as an effective and efficient tool for image restoration [92]. It has been successfully applied to both image denoising and non-blind image deconvolution, producing state-of-the-art results while maintaining high computational efficiency. Motivated by this success, we adopt the shrinkage field model for the challenging problem of *blind* deblurring of document images. In particular, we propose a multi-scale, interleaved CSF which estimates the unknown blur kernel while progressively refining the estimation of the latent image. This is also partly inspired by [90], which proposes an interleaved cascade of RTF to *post*-improve the results of state-of-the-art natural image deblurring methods. However, in contrast to [90], our method *does not* depend on an initial kernel estimation from an auxiliary method. Instead, we estimate both the unknown blur kernel and latent sharp image from a single blurry input image.

5.2.1 Cascade of shrinkage fields

The shrinkage field model can be derived from the FoE model [87]:

$$\operatorname*{argmin}_{\mathbf{x}} \mathcal{D}(\mathbf{x}, \mathbf{b}) + \sum\nolimits_{i=1}^{N} \rho_i(\mathbf{F}_i \mathbf{x}), \tag{5.2}$$

where $\mathcal{D}$ represents the data fidelity given measurement $\mathbf{b}$, matrix $\mathbf{F}_i$ represents the corresponding 2D convolution with filter $\mathbf{f}_i$, and ρ_i is the penalty on the filter response. Half-quadratic optimization [30], a popular approach for the optimization of common random field models, introduces auxiliary variables $\mathbf{u}_i$ for all filter responses $\mathbf{F}_i\mathbf{x}$ and replaces the energy optimization problem in Eq. 5.2 with a quadratic relaxation:

$$\operatorname*{argmin}_{\mathbf{x},\mathbf{u}} \mathcal{D}(\mathbf{x}, \mathbf{b}) + \sum\nolimits_{i=1}^{N} \left(\beta ||\mathbf{F}_i \mathbf{x} - \mathbf{u}_i||_2^2 + \rho_i(\mathbf{u}_i)\right), \tag{5.3}$$

which for $\beta \to \infty$ converges to the original problem in Eq. 5.2. The key insight of [92] is that the minimizer of the second term w.r.t. $\mathbf{u}_i$ can be replaced by a flexible 1D shrinkage function ψ_i of filter response $\mathbf{F}_i\mathbf{x}$.

Different from standard random fields which are parameterized through potential functions, SF models the shrinkage functions associated with the potential directly. Given data formation model as in Eq. 5.1, this reduces the original optimization problem Eq. 5.2 to a single quadratic minimization problem in each iteration, which can be solved efficiently as

$$\mathbf{x}^t = \mathcal{F}^{-1}\left[\frac{\mathcal{F}(\mathbf{K}_{t-1}^{\mathsf{T}}\mathbf{b} + \lambda^t \sum_{i=1}^{N} \mathbf{F}_i^{t\mathsf{T}} \psi_i^t(\mathbf{F}_i^t \mathbf{x}^{t-1}))}{\mathcal{F}(\mathbf{K}_{t-1}^{\mathsf{T}}) \cdot \mathcal{F}(\mathbf{K}_{t-1}) + \lambda^t \sum_{i=1}^{N} \mathcal{F}(\mathbf{F}_i^{t\mathsf{T}}) \cdot \mathcal{F}(\mathbf{F}_i^t)}\right], \tag{5.4}$$

where t is iteration index, $\mathbf{K}$ is the blur kernel matrix, $\mathcal{F}$ and $\mathcal{F}^{-1}$ indicate Fourier transform and its inverse, and ψ_i the shrinkage function. The model parameters $\Theta^t = (\mathbf{f}_i^t, \psi_i^t, \lambda^t)$ are trained by loss-minimization, e.g. by minimizing the ℓ_2 error between estimated images $\mathbf{x}^t$ and the ground truth. Performing multiple predictions of Eq. 5.4 is known as a cascade of shrinkage fields. For more details on the shrinkage fields model we refer readers to the supplemental material and [92].

5.2.2 Multi-scale interleaved CSF for blind deconvolution

We do not follow the commonly used two-step deblurring procedure where kernel estimation and final latent image recovery are separated. Instead, we learn an interleaved CSF that directly produces both the estimated blur kernel and the predicted latent image. Our interleaved CSF is obtained by stacking multiple SFs into a cascade that is intermitted by kernel refinement steps. This cascade generates a sequence of iteratively refined blur kernel and latent image estimates, i.e. $\{\mathbf{k}^t\}_{t=1,..,T}$ and $\{\mathbf{x}^t\}_{t=1,..,T}$ respectively. At each stage of the cascade, we employ a separately trained SF model for sharp image restoration. In addition, we learn an auxiliary SF model which generates a latent image $\mathbf{z}^t$ that is used to facilitate blur kernel estimation. The reason of including this extra SF model at each stage is to allow for selecting features that might benefit kernel estimation and eliminating other features and artifacts. Note that the idea of introducing such a latent feature image for improving kernel estimation is not new, and is a rather common practice in recent state-of-the-art blind deconvolution methods [20, 113]. Fig. 5.2 depicts a schematic illustration of a single stage of our interleaved CSF approach.

More specifically, given the input image $\mathbf{b}$, our method recovers $\mathbf{k}$ and $\mathbf{x}$ si-

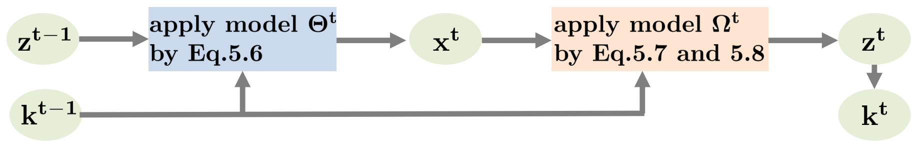

$\mathbf{z^{t-1}}$
apply model $\mathbf{\Theta^t}$ by Eq.5.6
$\mathbf{x^t}$
apply model $\mathbf{\Omega^t}$ by Eq.5.7 and 5.8
$\mathbf{z^t}$
$\mathbf{k^{t-1}}$
$\mathbf{k^t}$

Figure 5.2: Algorithm architecture.

multaneously by solving the following optimization problem:

$$\begin{aligned}(\mathbf{x},\mathbf{k}) &= \underset{\mathbf{x},\mathbf{k}}{\operatorname{argmin}} ||\mathbf{b}-\mathbf{k}\otimes\mathbf{x}||_2^2 + \sum\nolimits_{i=1}^{N} \rho_i(\mathbf{F}_i\mathbf{x}) + \tau||\mathbf{k}||_2^2, \\ &s.t. \quad \mathbf{k}\geq 0, ||\mathbf{k}||_1 = 1\end{aligned} \tag{5.5}$$

To this end, our proposed interleaved CSF alternates between the following blur kernel and latent image estimation steps:

Update $\mathbf{x}^t$. For sharp image update we train a SF model with parameters $\Theta^t = (\mathbf{f}_i^t, \psi_i^t, \lambda^t)$. Analogously to Eq. 5.4 we obtain the following update for $\mathbf{x}^t$ at iteration t:

$$\mathbf{x}^t = \mathcal{F}^{-1}\left[\frac{\mathcal{F}(\mathbf{K}_{t-1}^{\mathsf{T}}\mathbf{b} + \lambda^t \sum_{i=1}^{N} \mathbf{F}_i^{t\,\mathsf{T}} \psi_i^t(\mathbf{F}_i^t\mathbf{z}^{t-1}))}{\mathcal{F}(\mathbf{K}_{t-1}^{\mathsf{T}})\cdot\mathcal{F}(\mathbf{K}_{t-1}) + \lambda^t \sum_{i=1}^{N} \mathcal{F}(\mathbf{F}_i^{t\,\mathsf{T}})\cdot\mathcal{F}(\mathbf{F}_i^t)}\right] \tag{5.6}$$

Update $\mathbf{z}^t$ and $\mathbf{k}^t$. For kernel estimation we first update the latent image $\mathbf{z}^t$ from $\mathbf{x}^t$ by learning a separate SF model. Denoting convolution with filter $\mathbf{g}_i^t$ by matrix $\mathbf{G}_i^t$, we have:

$$\mathbf{z}^t = \mathcal{F}^{-1}\left[\frac{\mathcal{F}(\mathbf{K}_{t-1}^{\mathsf{T}}\mathbf{b} + \eta^t \sum_{i=1}^{N} \mathbf{G}_i^{t\,\mathsf{T}} \phi_i^t(\mathbf{G}_i^t\mathbf{x}^{t}))}{\mathcal{F}(\mathbf{K}_{t-1}^{\mathsf{T}})\cdot\mathcal{F}(\mathbf{K}_{t-1}) + \eta^t \sum_{i=1}^{N} \mathcal{F}(\mathbf{G}_i^{t\,\mathsf{T}})\cdot\mathcal{F}(\mathbf{G}_i^t)}\right] \tag{5.7}$$

For kernel estimation we employ a simple Tikhonov prior. Given the estimated latent image $\mathbf{z}^t$ and the blurry input image $\mathbf{b}$, the update for $\mathbf{k}^t$ reads:

$$\mathbf{k}^t = \mathcal{F}^{-1}\left[\frac{\mathcal{F}(\mathbf{z}^t)^* \cdot \mathcal{F}(\mathbf{b})}{\mathcal{F}(\mathbf{z}^t)^* \cdot \mathcal{F}(\mathbf{z}^t) + \tau^t}\right], \tag{5.8}$$

where $*$ indicates complex conjugate. The model parameters learned at this step are denoted as $\Omega^t = (\mathbf{g}_i^t, \phi_i^t, \eta^t, \tau^t)$. Note that Ω^t are trained to facilitate the update of both kernel $\mathbf{k}^t$ and image $\mathbf{z}^t$.

The $\mathbf{x}^t$ update step in Eq. 5.6 takes $\mathbf{z}^{t-1}$ rather than $\mathbf{x}^{t-1}$ as input, as $\mathbf{z}^{t-1}$ improves from $\mathbf{x}^{t-1}$ w.r.t. removing blur by Eq. 5.7 at iteration $t-1$. $\mathbf{x}^t$ and $\mathbf{z}^t$ is observed to converge as the latent image and kernel are recovered.

Algorithm 6 Blind deblurring at one scale

```
Input: blurry image $\mathbf{b}$
Output: estimated image $\mathbf{x}$ and kernel $\mathbf{k}$.
1: for $t = 1$ to 5 do
2:     Update $\mathbf{x}^t$ by Eq. 5.6.
3:     Update $\mathbf{z}^t$ by Eq. 5.7.
4:     Update $\mathbf{k}^t$ by Eq. 5.8.
5:     $\mathbf{k}^t = \max(0, \mathbf{k}^t), \mathbf{k}^t = \mathbf{k}^t / ||\mathbf{k}^t||_1$.
6: end for
```

Algorithm 6 summarizes the proposed approach for blind deblurring of document images. Note that there is translation and scaling ambiguity between the sharp image and blur kernel at blind deconvolution. The estimated kernel is normalized such that all its pixel values sum up to one. In Algorithm 7 for training, $\mathbf{x}^t$ is shifted to better align with the ground truth image $\bar{\mathbf{x}}$, before updating $\mathbf{k}$. We find that our algorithm usually converges in 5 iterations per scale.

5.2.3 Learning

Our interleaved CSF has two sets of model parameters at every stage $t = 1, .., 5$, one for sharp image restoration, $\Theta^t = (\mathbf{f}_i^t, \psi_i^t, \lambda^t)$, and the other for blur kernel estimation, $\Omega^t = (\mathbf{g}_i^t, \phi_i^t, \eta^t, \tau^t)$. All model parameters are learned through loss-minimization.

Note that in addition to the blurry input image, each model receives also the previous image and blur kernel predictions as input, which are progressively refined at each iteration. This is in contrast to the non-blind deconvolution setting of [92], where the blur kernel is known and is kept fixed throughout all stages. Our interleaved CSF model is trained in a greedy fashion, i.e. stage by stage such that

Algorithm 7 Learning at one scale

```
Input: blurry image $\mathbf{b}$; true image $\bar{\mathbf{x}}$; true kernel $\bar{\mathbf{k}}$.
Output: model parameters $(\mathbf{f}_i^t, \psi_i^t, \lambda^t, \mathbf{g}_i^t, \phi_i^t, \eta^t, \tau^t)$
1: for $t = 1$ to $5$ do
2:     Train model parameters: $(\mathbf{f}_i^t, \psi_i^t, \lambda^t)$ to minimize $||\mathbf{x}^t - \bar{\mathbf{x}}||_2^2$ with gradient given in Eq. 5.9.
3:     Update $\mathbf{x}^t$ by Eq. 5.6.
4:     Shift $\mathbf{x}^t$ to better align with $\bar{\mathbf{x}}$.
5:     Train model parameters: $(\mathbf{g}_i^t, \phi_i^t, \eta^t, \tau^t)$ to minimize $||\mathbf{k}^t - \bar{\mathbf{k}}||_2^2 + \alpha||\mathbf{z}^t - \bar{\mathbf{x}}||_2^2$ with gradient given in Eq. 5.10.
6:     Update $\mathbf{z}^t$ by Eq. 5.7.
7:     Update $\mathbf{k}^t$ by Eq. 5.8.
8:     $\mathbf{k}^t = \max(0, \mathbf{k}^t), \mathbf{k}^t = \mathbf{k}^t / ||\mathbf{k}^t||_1$.
9: end for
```

the learned SF models at one stage are able to adapt to the kernel and latent image estimated at the previous stage.

More specifically, at each stage we update our model parameters by iterating between the following two steps:

Update $\mathbf{x}^t$. To learn the model parameters Θ^t, we minimize the ℓ_2 error between the current image estimate and the ground truth image $\bar{\mathbf{x}}$, i.e. $\ell = ||\mathbf{x}^t - \bar{\mathbf{x}}||_2^2$. Its gradient w.r.t. the model parameters $\Theta^t = (\mathbf{f}_i^t, \psi_i^t, \lambda^t)$ can be readily computed as

$$\frac{\partial \ell}{\Theta^t} = \frac{\partial \mathbf{x}^t}{\partial \Theta^t} \frac{\partial \ell}{\mathbf{x}^t} \tag{5.9}$$

The derivatives for specific model parameters are omitted here for brevity, but can be found in Appendix A.2.

Update $\mathbf{z}^t$ and $\mathbf{k}^t$. The model parameters Ω^t of the SF models for kernel estimation at stage t are learned by minimizing the loss function $\ell = ||\mathbf{k}^t - \bar{\mathbf{k}}||_2^2 + \alpha||\mathbf{z}^t - \bar{\mathbf{x}}||_2^2$, where $\bar{\mathbf{k}}$ denotes the ground truth blur kernel and α is a coupling constant. This loss accounts for errors in the kernel but also prevents

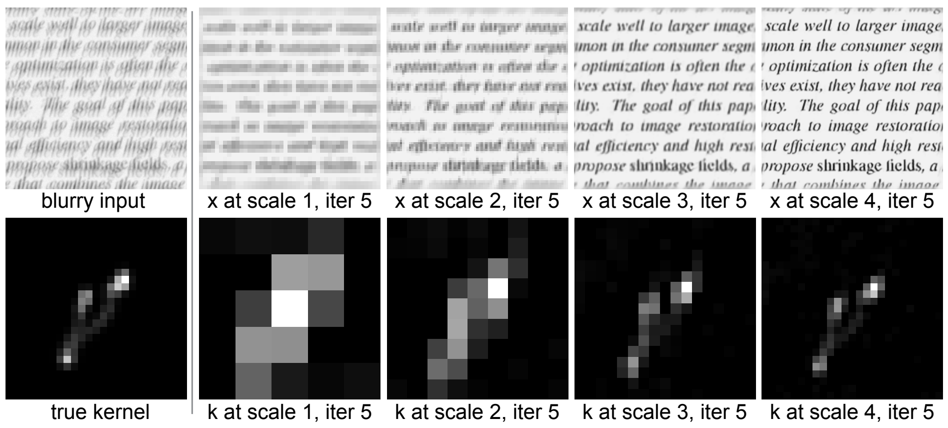


Figure 5.3: Example intermediate results of our algorithm on a synthetic test image.

the latent image used in Eq. (5.8) to diverge. Its gradient w.r.t. the model parameters $\Omega^t = (\mathbf{g}_i^t, \phi_i^t, \eta^t, \tau^t)$ reads

$$\frac{\partial \ell}{\partial \Omega^t} = \frac{\partial \mathbf{z}^t}{\partial \Omega^t}\frac{\partial \mathbf{k}^t}{\partial \mathbf{z}^t}\frac{\partial \ell}{\partial \mathbf{k}^t} + \frac{\partial \mathbf{k}^t}{\partial \Omega^t}\frac{\partial \ell}{\partial \mathbf{k}^t} + \frac{\partial \mathbf{z}^t}{\partial \Omega^t}\frac{\partial \ell}{\partial \mathbf{z}^t} \tag{5.10}$$

Again, details for the computation of the derivatives w.r.t. to specific model parameters are included in the supplemental material. We want to point out that the kernel estimation error $||\mathbf{k}^t - \bar{\mathbf{k}}||_2^2$ is back-propagated to the model parameters $(\mathbf{g}_i^t, \phi_i^t, \eta^t)$ in the SF for $\mathbf{z}^t$. Hence, the latent image $\mathbf{z}^t$ is tailored for accurate kernel estimation and predicted such that the refinement in $\mathbf{k}^t$ in each iteration is optimal. This differs from related work in [90, 119].

Multi-scale approach

Our algorithm uses a multi-scale approach to prevent bad local optima. The kernel widths that are used at different scales are 5, 9, 17, 25 pixels. At each scale s, the blurry image $\mathbf{b}^s$, the true latent image $\bar{\mathbf{x}}^s$ and $\bar{\mathbf{k}}^s$ are downsampled (and normalized for $\bar{\mathbf{k}}^s$) from their original resolution. The scale index s is omitted for convenience. At the beginning of each scale $s > 1$, the estimated image $\mathbf{x}$ is initialized by bicubic upsampling its estimation at the previous scale, and the blur kernel $\mathbf{k}$ is initialized

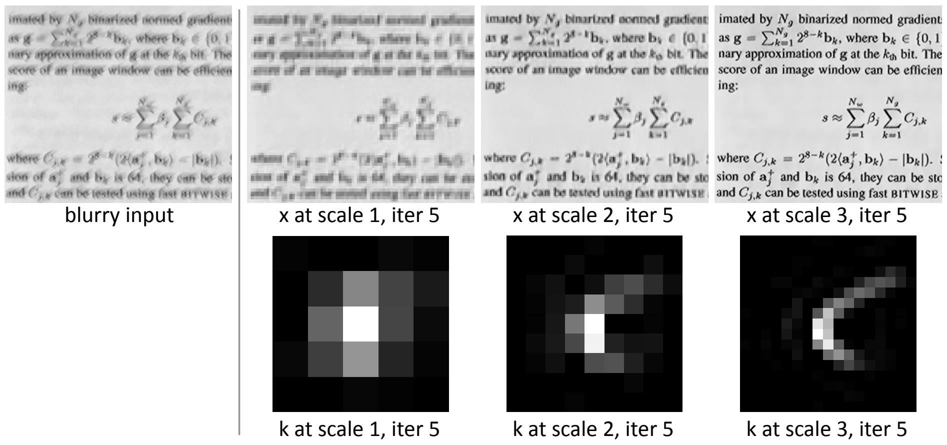
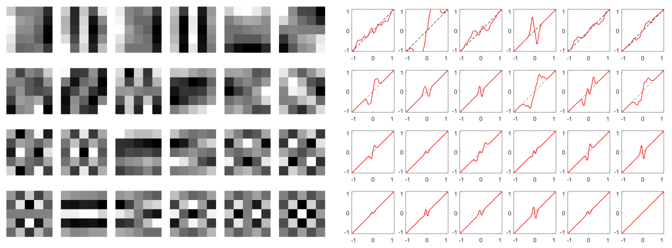


Figure 5.4: Example intermediate results (cropped) of our algorithm on a real-world test image (shown in Fig.5.8 in Chapter 5).

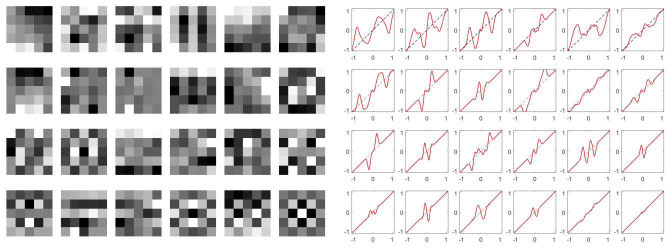
by nearest-neighbor upsampling, followed by re-normalization. At the coarsest scale $s = 1$, $\mathbf{x}$ is initialized as $\mathbf{b}$ and $\mathbf{k}$ is initialized as a delta peak. The coupling constant α in kernel estimation loss is defined as $\alpha = r \cdot \eta$, where r is the ratio between pixel numbers in kernel $\mathbf{k}^t$ and image $\mathbf{z}^t$ at current scale, η is initialized with 1 at the coarsest scale and at each subsequent scale it is multiplied by a factor of 0.25. Algorithm 7 summarizes our learning procedure for a single scale of our CSF model. Fig. 5.3 and 5.4 shows intermediate results of our estimated image $\mathbf{x}$ and kernel $\mathbf{k}$ at each scale. Note that our algorithm simultaneously estimates the latent image and blur kernel, and does not need extra non-blind deconvolution steps.

Model complexity

In both the model Θ^t for $\mathbf{x}^t$ and model Ω^t for ($\mathbf{z}^t$, $\mathbf{k}^t$), we choose to use 24 filters $\mathbf{f}_i^t$ of size 5×5 for trade-off between result quality, model complexity and time efficiency. As in [92], we initialize the filters with a DCT filter bank. Each shrinkage function ψ_i^t and ϕ_i^t are composed of 51 equidistant-positioned radial basis functions (RBFs) and are initialized as identity function. We further enforce central symmetry to the shrinkage functions, so that the number of trainable RBFs reduces



(**a**) Learned filters $\mathbf{f}_i^t$ and shrinkage functions ψ_i^t in Eq. 5.6.



(**b**) Learned filters $\mathbf{g}_i^t$ and shrinkage functions ϕ_i^t in Eq. 5.7.

Figure 5.5: Learned filters and shrinkage functions (at 3rd scale, 1st iteration) for updating $\mathbf{x}^t$ (Eq. 5.6) and $\mathbf{z}^t$, $\mathbf{k}^t$ (Eq. 5.7), respectively. Other parameters learned at this iteration: λ^t=0.5757, η^t=0.0218, τ^t=0.0018.

by half to 25. Fig. 5.5 visualizes some learned models.

Training datasets

We have found that that our method works well with a relatively small training dataset without over-fitting. We collected 20 motion blur kernels from [92], and randomly rotated them to generate 60 different kernels. We collected 60 sharp patches of 250×250 pixels cropped from documents rendered around 175 PPI, and rotated each with a random angle between -4 and 4 degrees. We then generated 60 blurry images by convolving each pair of sharp image and kernel, followed by adding white Gaussian noise and quantizing to 8 bits. Fig. 5.6 visualizes example blur kernels and images used at our training. We used the L-BFGS solver [91] in Matlab for training, which took about 12 hours on a desktop with an Intel Xeon CPU.

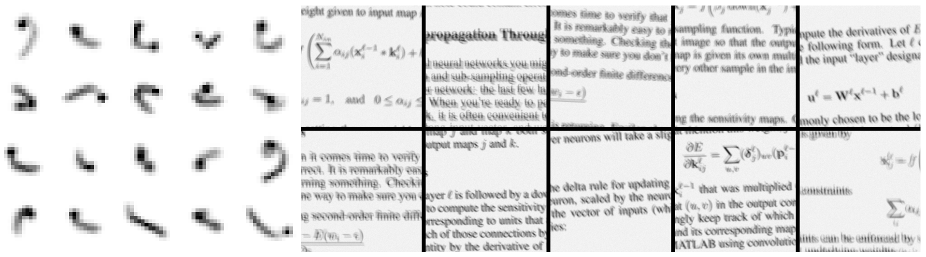


Figure 5.6: Example kernels and images used at training of our algorithm.

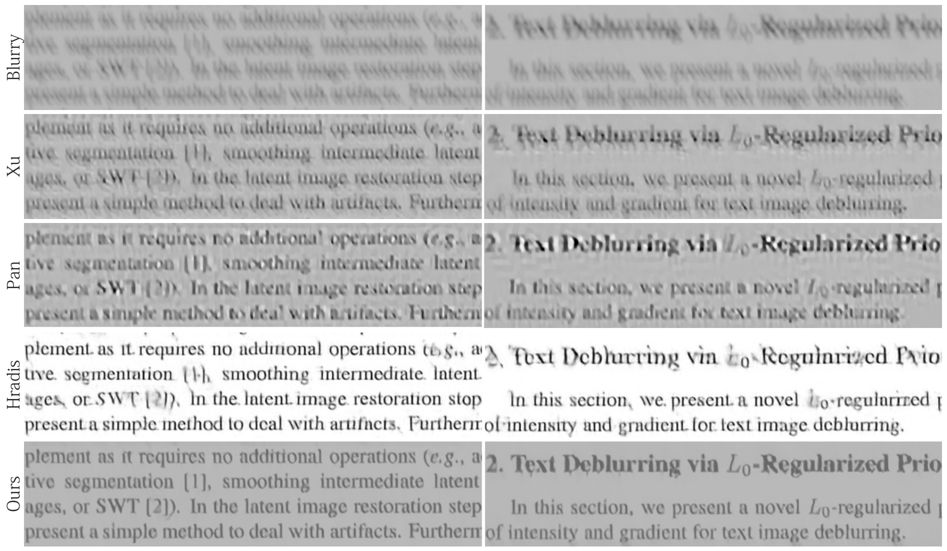


Figure 5.7: Comparison on a real image taken from [48]. Row 1-5 from top to bottom show the blurry image, result of Xu [1], Pan [78], Hradiš [48] and our method. Two cropped regions are shown here, the full resolution results along with more examples can be found in the supplemental.

5.3 Results

In this section we evaluate the proposed algorithm on both synthetic and real-world images. We compare with Pan [78] and Hradiš [48], the state-of-the-art methods for text image blind deblurring, and the natural image deblurring software produced by Xu [1], which are based on recently proposed state-of-the-art techniques [113, 114]. We used the code and binaries provided by the authors and tuned the parameters to generate the best possible results.

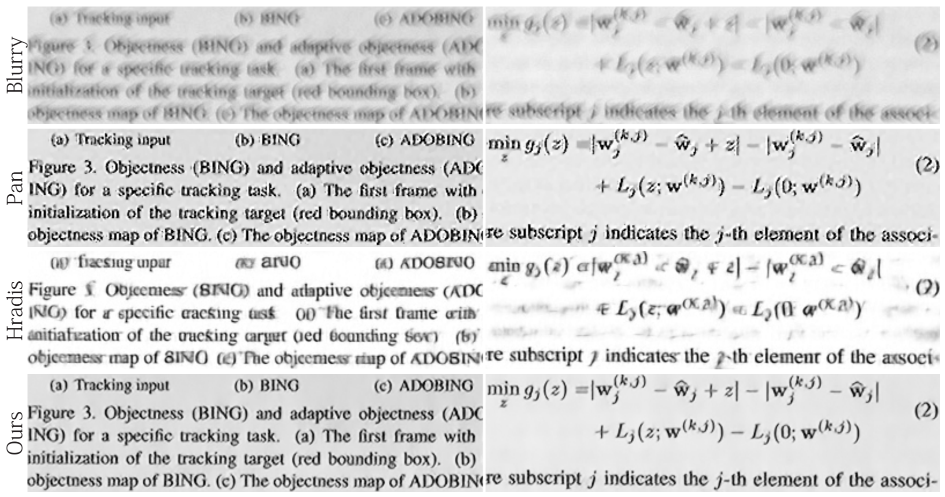


Figure 5.8: Comparison on a real image taken from [48]. Row 1-4 from top to bottom show the blurry image, result of Pan [78], Hradiš [48] and our method. Two cropped regions are shown, the full resolution results along with more results can be found in the supplemental.

Real-world images

In Fig. 5.7 and 5.8 we show comparisons on real images. The result images of Xu [1] and Pan [78] contain obvious artifacts due to ineffective image priors that lead to inaccurate kernel estimation. Hradiš [48] fails to recover many characters and distorted the font type and illumination. Our method produces the best results in these cases, and our results are both visually pleasing and highly legible. The full resolution images and more results can be found on the author webpage.

Quantitative comparisons

For quantitative evaluation, we test all methods on a synthetic dataset and compare results in terms of PSNR. We collect 8 sharp document images with 250×250 pixels cropped from documents rendered at 150 PPI (similar PPI as used for training in [48]). Each image is blurred with 8 kernels at 25×25 collected from [63], followed by adding 1% Gaussian noise and 8-bit quantization. In Fig. 5.9, we show the average PSNR values of all 8 test images synthesized with the same blur kernel. Our method outperforms other methods in all cases by 0.5-6.0 dB. Hradiš [48] has

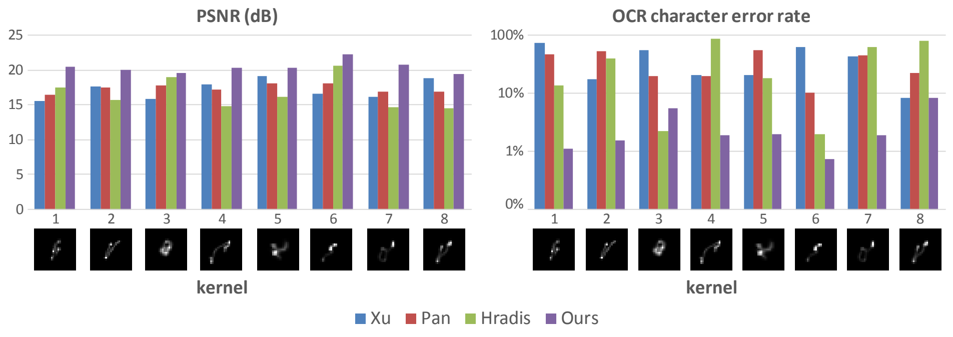
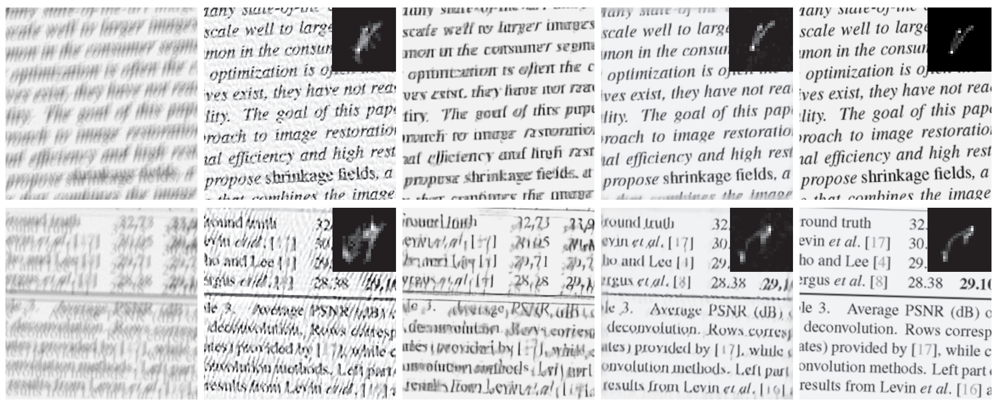


Figure 5.9: PSNR and OCR comparison on a synthetic test dataset with 8 blur kernels.



(a) Blurry **(b)** Pan [78] **(c)** Hradiš[48] **(d)** Ours **(e)** GT

Figure 5.10: Comparison on synthetic images from the PSNR experiments in Fig. 5.9. Note that the original results of [48] break the illumination of the images. We clamp the intensity of their results to match the ground truth image before computing the PSNR values.

close performance to ours on kernel #3, which is close to defocus blur. It also performs reasonably well on kernel #6 which features a simple motion path, but fails on other more challenging kernels. Some results along with the estimated kernels are shown in Fig. 5.10 for visual comparison.

An interesting question one may ask is whether improved deblur can directly lead to better OCR accuracy. To answer this question we evaluate OCR accuracy using the software ABBYY FineReader 12. We collected 8 sharp document images

Table 5.1: Run-time comparison (in seconds).

Image size	256^2	512^2	1024^2
Xu [1] (C++)	14.8	33.4	-
Pan [78] (Matlab)	19.6	84.3	271.9
Hradiš [48] (C++)	48.5	193.7	594.9
Hradiš [48] (GPU)	0.3	1.0	3.1
Ours (Matlab)	2.0	3.9	11.4
Pre-computation (Matlab)	1.8	4.6	15.3

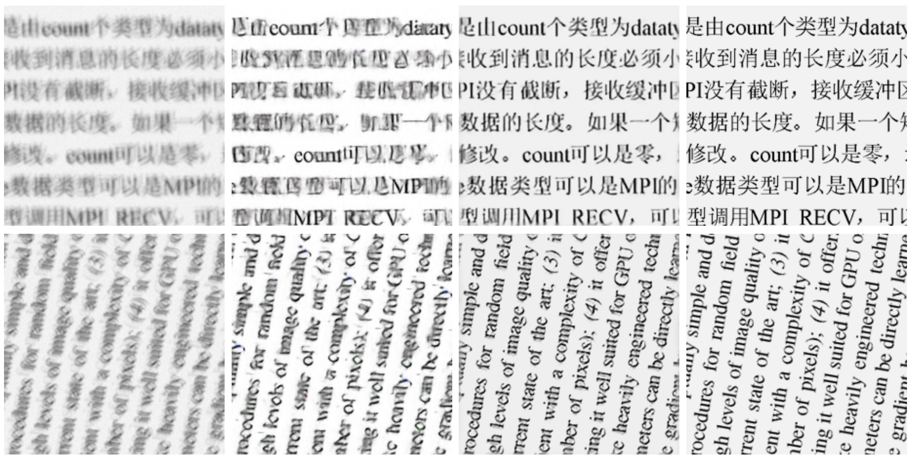
from the OCR test dataset in [48]. Each document image contains a continuous paragraph. We synthesized 64 blurry images with the 8 kernels and 1% Gaussian noise similarly as in the PSNR comparison. We run the OCR software and used the script provided by [48] to compute the average character error rate for all 8 test images synthesized with the same kernel[1]. The results are shown in Fig. 5.9. They are consistent with the PSNR results also in Fig. 5.9. Hradiš [48] performs well on kernel #3 and #6 but fails on other challenging kernels, while our method is consistently better than others. All the test images and results for PSNR and OCR comparisons are included in the supplemental material.

Run-time comparison

Table 5.1 provides a comparison on computational efficiency, using images blurred by a 17×17 kernel at three different resolutions. The experiments were done on an Intel i7 CPU with 16GB RAM and a GeForce GTX TITAN GPU. Assuming the image sensor resolution is a known priori[2], we pre-compute the FFTs of the trained filters $\mathbf{f}_i$ and $\mathbf{g}_i$ for maximal efficiency. We report the timing of our Matlab implementation on CPU. A GPU implementation should significantly reduce the time as our method only requires FFT, 2D convolution and 1D look-up-table (LUT) operations, which is our future work.

[1]We used the script "eval.py" downloaded from the author webpage [48] to compute the error rate.

[2]This is a common assumption especially for batch processing of document images.



(a) Blurry **(b)** Hradiš[48] **(c)** Ours **(d)** GT

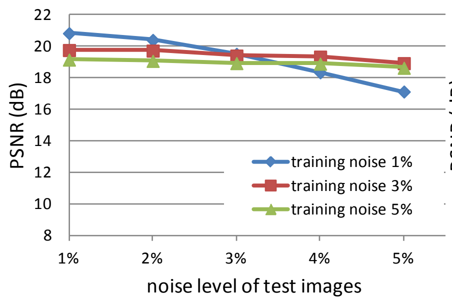
Figure 5.11: Comparison on non-English text and severely rotated images. Note that such non-English text and large rotation were not included in our training dataset.

Robustness

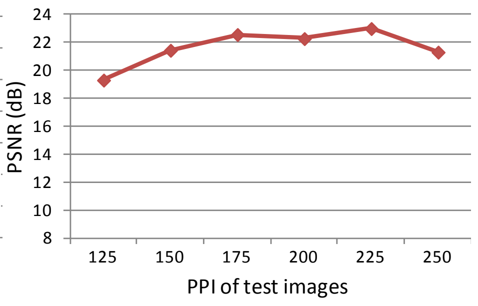
In Fig. 5.11, we show results on non-English text and severely rotated image. Although both Hradiš [48] and our method are only trained on English text data, our method can be applied to non-English text as well. This is a great benefit of our method as we do not need to train on every different language, or increase the model complexity to handle them as [48] would need to do. Our method is also robust against a significant change of page orientation, which cannot be handled well by [48].

In Fig. 5.12, we show the results of our method when the noise level and PPI of the test data differs from the training data. Fig. 5.12(a) shows that the performance of our method is fairly steady when the noise level in the test images is not too much higher than that of the training data, meaning that the models trained at sparse noise levels are sufficient for practical use. Fig. 5.12(b) shows that our method works well in a fairly broad range of image PPIs given the training data are around 175 PPI.

In Fig. 5.13, we show a comparison on a real image with large-font text. Fol-

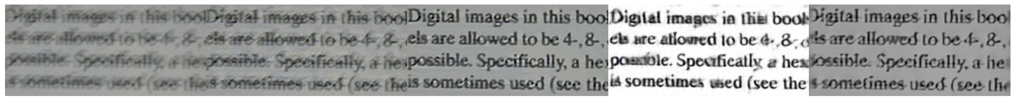


(a)



(b)

Figure 5.12: Robustness test on noise level and image PPI.



(a) Blurry **(b)** Xu [1] **(c)** Pan [78] **(d)** Hradiš[48] **(e)** Ours

Figure 5.13: Comparison on a real image with large-font text. The reference results are from [48]. Following [48], the input of (d) Hradiš' and (e) our method was downsampled by factor of 3.

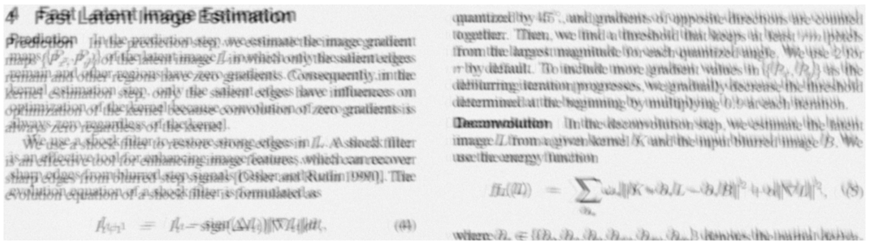
lowing [48], the input of Hradiš' and our method was downsampled by factor of 3 in order to apply the trained models without re-training. Although such downsampling breaks the image formation model in Eq. 5.1, our method can still generate reasonable result.

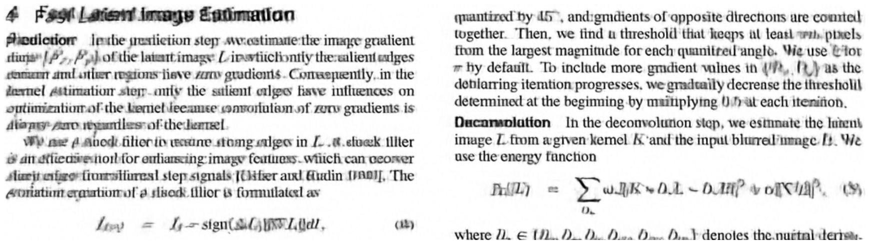
Non-uniform blur

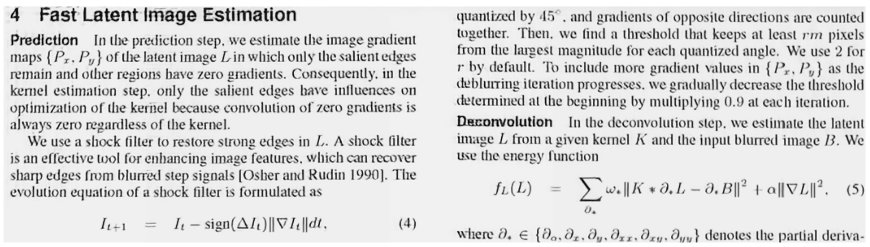
Our method can be easily extended to handle non-uniform blur by dividing the image into overlapped tiles, deblurring each tile with our proposed algorithm, and then realigning the resulting tiles to generate the final estimated image. An example is shown in Fig. 5.14.

Documents with color figures

Our algorithm can be easily extended to handle the documents with color figures. We first run the blind deblurring algorithm on text regions to recover the latent text


(**a**) Blurry


(**b**) Hradiš [48]



4 Fast Latent Image Estimation

Prediction In the prediction step, we estimate the image gradient maps $\{P_x, P_y\}$ of the latent image L in which only the salient edges remain and other regions have zero gradients. Consequently, in the kernel estimation step, only the salient edges have influences on optimization of the kernel because convolution of zero gradients is always zero regardless of the kernel.

We use a shock filter to restore strong edges in L. A shock filter is an effective tool for enhancing image features, which can recover sharp edges from blurred step signals [Osher and Rudin 1990]. The evolution equation of a shock filter is formulated as

$$I_{t+1} = I_t - \text{sign}(\Delta I_t)\|\nabla I_t\|dt, \quad (4)$$

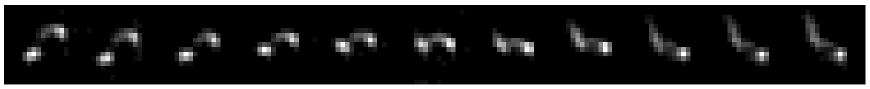
quantized by 45°, and gradients of opposite directions are counted together. Then, we find a threshold that keeps at least rm pixels from the largest magnitude for each quantized angle. We use 2 for r by default. To include more gradient values in $\{P_x, P_y\}$ as the deblurring iteration progresses, we gradually decrease the threshold determined at the beginning by multiplying 0.9 at each iteration.

Deconvolution In the deconvolution step, we estimate the latent image L from a given kernel K and the input blurred image B. We use the energy function

$$f_L(L) = \sum_{\partial_*} \omega_* \|K * \partial_* L - \partial_* B\|^2 + \alpha\|\nabla L\|^2, \quad (5)$$

where $\partial_* \in \{\partial_o, \partial_x, \partial_y, \partial_{xx}, \partial_{xy}, \partial_{yy}\}$ denotes the partial deriva-

(**c**) Ours


(**d**) Our estimated kernel

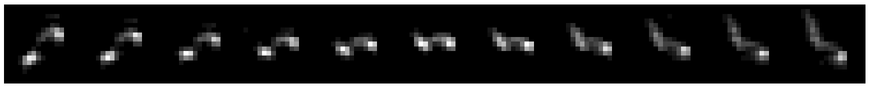

(**e**) Ground truth kernel

Figure 5.14: Results on spatially-varying blur kernels. The blurry input is synthesized with the EFF model [47] to approximate practical pixel-wise variant blur.

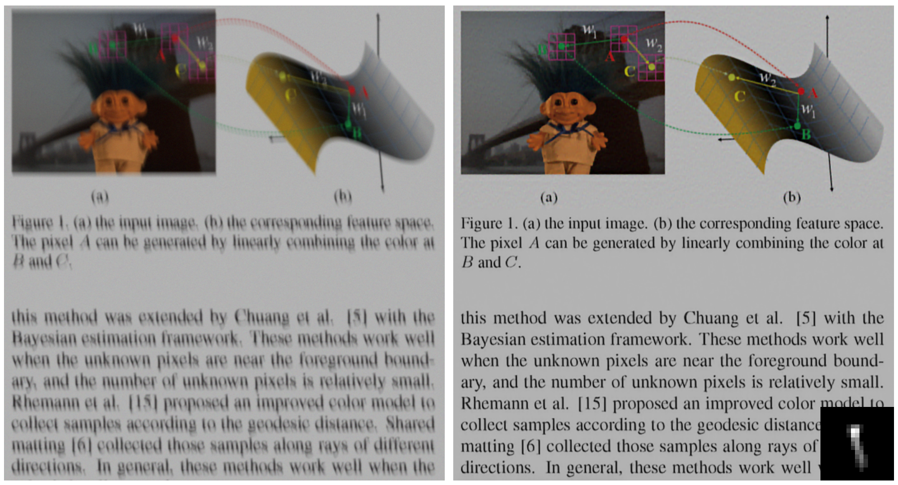


Figure 1. (a) the input image. (b) the corresponding feature space. The pixel A can be generated by linearly combining the color at B and C.

this method was extended by Chuang et al. [5] with the Bayesian estimation framework. These methods work well when the unknown pixels are near the foreground boundary, and the number of unknown pixels is relatively small. Rhemann et al. [15] proposed an improved color model to collect samples according to the geodesic distance. Shared matting [6] collected those samples along rays of different directions. In general, these methods work well when the

(a) Blurry input **(b)** Our deblurred image

Figure 5.15: Result on example document containing color figures. The right-bottom corner in (b) shows the blur kernel estimated from the text regions. In this example we simply use [58] for the non-blind deblurring step, although [92] can be used instead for improved results.

images and blur kernels. Then we take the kernels estimated from the text regions around the color figures (optionally interpolate the kernels using the Efficient Filter Flow (EFF) method [47] for spatially-varying blur), and deblur the color figure regions as non-blind deconvolution. Finally, we align the text and figure regions to generate the final image. An example is shown in in Fig. 5.15.

It is a benefit that our algorithm jointly estimates the latent image and the blur kernel, and the latter can be further used for deblurring non-text regions non-blindly. Hradiš *et al* [48] does not recover the blur kernel thus cannot handle the figure regions in the document.

5.4 Conclusion and future work

In this chapter we present a new algorithm for fast and high-quality blind deconvolution of document photographs. Our key idea is to to use high-order filters for document image regularization, and propose to learn such filters and influences from training data using multi-scale, interleaved cascade of shrinkage field models. Extensive experiments demonstrate that our approach not only produces higher quality results than the state-of-the-art methods, but is also computational efficient, and robust against noise level, language and page orientation changes that are not included in the training data.

Our method also has some limitations. It cannot fully recover the details of an image if it is degraded by large out-of-focus blur. In such case, Hradiš [48] may outperform our method given its excellent synthesis ability. As future work it would be interesting to combine both approaches. Although we only show learning our model on document photographs, we believe such a framework can also be applied to other domain-specific images, which we plan to explore in the future.

Chapter 6

Learning Proximal Operators for Image Restoration

6.1 Introduction

Recently several discriminative learning approaches [18, 92] have been proposed for effective image restoration, achieving convincing trade-off between image quality and computational efficiency. However, these methods require separate training for each restoration task and problem condition. This makes it time-consuming and difficult to encompass all tasks and conditions during training. In this chapter, we combine discriminative learning technique with formal optimization methods to learn generic priors that truly share across problem domains. Our models require a single-pass training and allow reuse across various problems and conditions while achieving comparable efficiency as previous discriminative approaches.

In particular, we make the following contributions:

- We propose proximal fields as a convolutional model for image priors that are computationally cheap to train and are shared across different image restoration tasks and problem conditions.
- Proximal fields are formulated as proximal operators, allowing their use in advanced proximal optimization algorithms.

- We show that our approach is general by demonstrating proximal fields for diverse low-level problems, such as denoising, deconvolution and inpainting, for varying noise settings.
- We show that our method can naturally be combined with existing likelihood and priors after being trained, to cover unseen tasks and to further improve reconstruction quality.

6.2 Method

The seminal work of FoE [87] generalizes the form of filter response based regularizers in the objective function given in Eq. 6.1. Vector $\mathbf{b}$ and $\mathbf{x}$ represents the observed and latent (desired) image respectively, matrix $\mathbf{A}$ is the sensing operator, matrix $\mathbf{F}_i$ represents 2D convolution with filter $\mathbf{f}_i$, and function ϕ_i represents the penalty on corresponding filter response $\mathbf{F}_i\mathbf{x}$. The positive scalar λ controls relative weight between the data fidelity (likelihood) and the regularization term.

$$\frac{\lambda}{2}||\mathbf{b} - \mathbf{A}\mathbf{x}||_2^2 + \sum_{i=1}^{N} \phi_i(\mathbf{F}_i\mathbf{x}) \tag{6.1}$$

The well-known anisotropic total-variation regularizer can be viewed as a special case of the FoE model where $\mathbf{f}_i$ is the derivative operator ∇ and ϕ_i the ℓ_1 norm.

It is difficult to directly minimize Eq. 6.1 when the penalty function ϕ_i is non-linear and/or non-smooth (e.g., ℓ_p norm, $0 < p \leq 1$). Proximal algorithms [10, 15, 30] instead, relax Eq. 6.1 and split the original problem into several easier subproblems that are solved alternately until convergence.

In this paper we employ the HQS algorithm [30] to relax Eq. 6.1, as it typically requires much fewer iterations to converge compared with other proximal methods such as ADMM [10] and PD [15]. The relaxed objective function is given in Eq. 6.2:

$$\frac{\lambda}{2}||\mathbf{b} - \mathbf{A}\mathbf{x}||_2^2 + \frac{\rho}{2}||\mathbf{z} - \mathbf{x}||_2^2 + \sum_{i=1}^{N} \phi_i(\mathbf{F}_i\mathbf{z}), \tag{6.2}$$

where a slack variable $\mathbf{z}$ is introduced to approximate $\mathbf{x}$, and ρ is a positive scalar.

While most related approaches [58, 92] relax Eq. 6.1 by splitting on $\mathbf{F}_i\mathbf{x}$ rather than $\mathbf{x}$, it would limit the model flexibility in our method. This will be explained more clearly in the next sections.

With the HQS algorithm, Eq. 6.2 is iteratively minimized by solving for the slack variable $\mathbf{z}$ and the latent image $\mathbf{x}$ alternately as in Eq. 6.3 and 6.4 ($t = 1, 2, ..., T$).

$$\mathbf{z}^t = \underset{\mathbf{z}}{\operatorname{argmin}} \left(\frac{\rho^t}{2} ||\mathbf{z} - \mathbf{x}^{t-1}||_2^2 + \sum_{i=1}^{N} \phi_i(\mathbf{F}_i\mathbf{z}) \right), \tag{6.3}$$

$$\mathbf{x}^t = \underset{\mathbf{x}}{\operatorname{argmin}} \left(\lambda ||\mathbf{b} - \mathbf{A}\mathbf{x}||_2^2 + \rho^t ||\mathbf{z}^t - \mathbf{x}||_2^2 \right), \tag{6.4}$$

where ρ^t increases as the iteration continues. The latter forces $\mathbf{z}$ to become an increasingly good approximation of $\mathbf{x}$, thus making Eq. 6.2 an increasingly good proxy for Eq. 6.1.

6.2.1 Proximal fields

The $\mathbf{z}^t$-update step in Eq. 6.3 can be viewed as a proximal operation:

$$\mathbf{z}^t := \mathbf{prox}_\Theta(\mathbf{x}^{t-1}, \rho^t), \tag{6.5}$$

where $\mathbf{prox}_\Theta$ is called proximal operator with model parameters Θ, which includes a number of filters $\mathbf{f}_i$ and corresponding penalty functions ϕ_i. To distinguish it from traditional proximal operators which typically contain a single filter, we call our generalized proximal operators $\mathbf{prox}_\Theta$ as *proximal fields*.

Inspired by the state-of-the-art discriminative methods [18, 92], we propose to learn the proximal fields model $\mathbf{prox}_\Theta$ and the weight scalar λ from training data. With the above HQS relaxation, the image prior and data-fidelity term in the original objective (Eq. 6.1) are contained in two separate subproblems (Eq. 6.3 and 6.4). This makes it possible to train an ensemble of diverse tasks (e.g., denoising, deblurring, or with different noise levels) each of which has its own data fidelity term and weight λ, while learning a single prior model $\mathbf{prox}_\Theta$ that is shared across ensemble tasks. This is in contrast to state-of-the-art discriminative methods such as CSF [92] and TRD [18] which train separate priors for each task.

The proximal operator $\mathbf{prox}_{\Theta}(\mathbf{x}^{t-1}, \rho^t)$ can be interpreted as processing $\mathbf{x}^{t-1}$ corrupted by zero-mean Gaussian noise. With this interpretation, we propose to define $\mathbf{prox}_{\Theta}(\mathbf{x}^{t-1})$ as a multi-stage non-linear diffusion process modified from the TRD [18] model, as given in Eq. 6.6.

$$\begin{aligned} \mathbf{z}_k^t &= \mathbf{z}_{k-1}^t - \sum_{i=1}^{N} \mathbf{F}_i^{k\mathsf{T}} \psi_i^k(\mathbf{F}_i^k \mathbf{z}_{k-1}^t), \\ s.t. \quad \mathbf{z}_0^t &= \mathbf{x}^{t-1}, \quad k = 1, 2, ..., K. \end{aligned} \tag{6.6}$$

where k is the stage index, filters $\mathbf{F}_i^k$, function ψ_i^k are trainable model parameters at each stage, and $\mathbf{z}_0^t$ is the initial value of $\mathbf{z}_k^t$. Note that, different from TRD, our model does not contain the reaction term which would be $-\rho^t \alpha_k(\mathbf{z}_{k-1}^t - \mathbf{x}^{t-1})$ with step size α_k. The main reasons for this modification are:

- The data constraint is contained in the $\mathbf{x}^t$ update in Eq. 6.4;
- More importantly, our model gets rid of the weight ρ^t which changes at each HQS iteration. Therefore, our proximal operator $\mathbf{prox}_{\Theta}(\mathbf{x}^{t-1}, \rho^t)$ is simplified to be:

$$\mathbf{z}^t := \mathbf{prox}_{\Theta}(\mathbf{x}^{t-1}) \tag{6.7}$$

Note that our proximal fields model $\Theta = \{\mathbf{F}_i^k, \psi_i^k | k \in \{1, \ldots, K\}\}$ is re-used at all HQS iteration t, making it different than previous discriminative methods (CSF, TRD). The parameters to learn in our method Ω includes λ's for each problem class p (restoration task and problem condition), and the proximal fields model Θ shared across different classes, i.e., $\Omega = \{\lambda_p, \Theta\}$. Even though the scalar parameters λ_p are trained, our method allows users to adjust them at test time for best performance and non-trained problem classes. This contrasts to previous discriminative approaches whose model parameters are fixed at test time. The subscript p indicating the problem class in λ_p is omitted below for convenience. The values of ρ^t are pre-selected: $\rho^1 = 1$ and $\rho^t = 2\rho^{t-1}$ for $t > 1$.

Note that a multi-stage model as in Eq. 6.6 is not possible if we split on $\mathbf{F}_i\mathbf{x}$ instead of $\mathbf{x}$ in Eq. 6.1 and 6.2. For clarity, an overview of the proposed algorithm

Algorithm 8 Proposed algorithm

```
Input: degraded image $\mathbf{b}$, weight $\lambda$ (optional)
Output: recovered image $\mathbf{x}$
 1: $\mathbf{x}^0 = \mathbf{b}, \rho^1 = 1$ (initialization)
 2: for $t = 1$ to $T$ do
 3:     (Update $\mathbf{z}^t$ by Eq. 6.6 below)
 4:     $\mathbf{z}_0^t = \mathbf{x}^{t-1}$
 5:     for $k = 1$ to $K$ do
 6:         $\mathbf{z}_k^t = \mathbf{z}_{k-1}^t - \sum_{i=1}^N {\mathbf{F}_i^k}^\mathsf{T} \psi_i^k(\mathbf{F}_i^k \mathbf{z}_{k-1}^t)$
 7:     end for
 8:     $\mathbf{z}^t = \mathbf{z}_K^t$
 9:     (Update $\mathbf{x}^t$ by Eq. 6.4 below)
10:     $\mathbf{x}^t = \operatorname{argmin}_\mathbf{x} \lambda||\mathbf{b} - \mathbf{A}\mathbf{x}||_2^2 + \rho^t||\mathbf{z}^t - \mathbf{x}||_2^2$
11:     $\rho^{t+1} = 2\rho^t$
12: end for
```

is given in Algorithm 8.

6.2.2 Learning

We consider denoising and deconvolution tasks at training, where the sensing operator $\mathbf{A}$ is an identity matrix, or a block circulant matrix with circulant blocks that represents 2D convolution with blur kernels respectively. In denoising tasks, $\mathbf{x}^t$ update in Eq. 6.4 has closed-form solution:

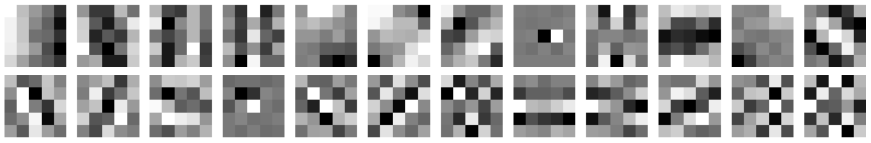
$$\mathbf{x}^t = \frac{\lambda \mathbf{b} + \rho^t \mathbf{z}^t}{\lambda + \rho^t} \tag{6.8}$$

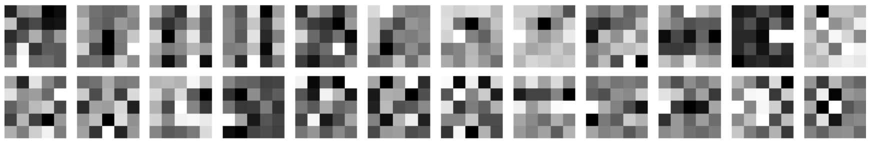
In deconvolution tasks, $\mathbf{x}^t$ update in Eq. 6.4 has closed-form solution in Fourier domain:

$$\mathbf{x}^t = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(\lambda \mathbf{A}^\mathsf{T}\mathbf{b} + \rho^t \mathbf{z}^t)}{\mathcal{F}(\lambda \mathbf{A}^\mathsf{T}\mathbf{A} + \rho^t)}\right), \tag{6.9}$$

where $\mathcal{F}$ and $\mathcal{F}^{-1}$ represent Fourier and inverse Fourier transform respectively. Note that, different than CSF [92], our method does not require FFT computation for denoising tasks.

We use L-BFGS solver [91] with analytic gradients for training. The training


(a) Filters $\mathbf{f}_i^1$ at stage 1.


(b) Filters $\mathbf{f}_i^2$ at stage 2.

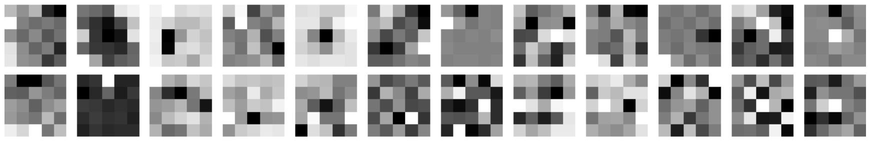

(c) Filters $\mathbf{f}_i^3$ at stage 3.

Figure 6.1: Trained filters at each stage (k in Eq. 6.6) of the proximal operator $\mathbf{prox}_\Theta$ in our model (3 stages each with 24 5×5 filters).

loss function ℓ is defined as the average PSNR of reconstructed images. The gradient of ℓ w.r.t. the model parameters $\Omega = \{\lambda_p, \Theta\}$ is computed by accumulating gradients at all HQS iterations, as shown in Eq. 6.10.

$$\frac{\partial \ell}{\partial \Omega} = \sum_{t=1}^{T} \left(\frac{\partial \mathbf{x}^t}{\partial \lambda} \frac{\partial \ell}{\partial \mathbf{x}^t} + \frac{\partial \mathbf{z}^t}{\partial \Theta} \left(\frac{\partial \ell}{\partial \mathbf{z}^t} + \frac{\partial \mathbf{x}^t}{\partial \mathbf{z}^t} \frac{\partial \ell}{\partial \mathbf{x}^t} \right) \right), \tag{6.10}$$

where the back-propagation inside our proximal operator is computed as:

$$\frac{\partial \mathbf{z}^t}{\partial \Theta} = \sum_{k=1}^{K} \frac{\partial \mathbf{z}_k^t}{\partial \Theta} \frac{\partial \mathbf{z}^t}{\partial \mathbf{z}_k^t} \tag{6.11}$$

The 1D functions ψ_i^k in Eq. 6.6 are parameterized as a linear combination of equidistant-positioned Gaussian kernels whose weights are trainable. More details of the derivation of analytic gradients can be found in Appendix A.3.

Progressive training

A progressive scheme is proposed to make the training more effectively. First, we set the number of HQS iterations to be 1, and train $\lambda's$ and the model Θ of each stage in $\mathbf{prox}_\Theta$ in a greedy fashion. Then, we gradually increase the number of HQS iterations from 1 to T where at each step the model $\Omega = \{\lambda, \Theta\}$ is refined from the result by previous step. The L-BFGS iterations are set to be 200 for the greedy training steps, and 100 for the refining steps. Fig. 6.1 shows examples of our learned filters in $\mathbf{prox}_\Theta$.

6.3 Results

Denoising

We compare our method with state-of-the-art image denoising techniques, including KSVD [26], FoE [87], BM3D [22], LSSC [71], WNNM [37], EPLL [118], opt-MRF [17], ARF [8], CSF [92] and TRD [18]. Our method is denoted in short as “ProxF”. The subscript in CSF$_5$ and TRD$_5$ indicates the number of cascaded stages (each stage has different model parameters). The subscript and superscript in ProxF$_3^5$ indicate the number of diffusion stages ($K = 3$ in Algorithm 8) in our proximal operator $\mathbf{prox}_\Theta$, and the number of HQS iterations ($T = 5$ in Algorithm 8), respectively. Note that the complexity (size) of our model is linear to K, but independent of T. CSF, TRD and ProxF use 24 filters of size 5×5 pixels at all stages in this section.

The compared discriminative methods, CSF$_5$ and TRD$_5$ both are trained at single noise level $\sigma = 15$ that is the same as the test images. In contrast, our model is trained on 400 images (100×100 pixels) cropped from [87] with random and discrete noise levels (standard deviation σ) varying between 5 and 25. The images with the same noise level share the same data fidelity weight λ at training.

All the methods are evaluated on the 68 test images with noise level $\sigma = 15$ from [87] and the averaged PSNR values are reported in Table 6.1. Our results are comparable to generic methods such as KSVD, FoE and BM3D , and very close to discriminative methods such as CSF$_5$, while at the same time being much more efficient which is demonstrated later. Besides, we simply use λ learned for images with noise $\sigma = 15$ at training to generate all the test results, although adjusting its value at test time can expectedly improve our results. Note that the compared discriminative methods (CSF, TRD) do not allow for such parameter tuning.

Table 6.1: Average PSNR (dB) on 68 images from [87] for image denoising.

KSVD	FoE	BM3D	LSSC	WNNM	EPLL
30.87	30.99	31.08	31.27	31.37	31.19
opt-MRF	ARF	CSF_5	TRD_5	ProxF_3^3	ProxF_3^5
31.18	30.70	31.14	31.30	30.91	31.00

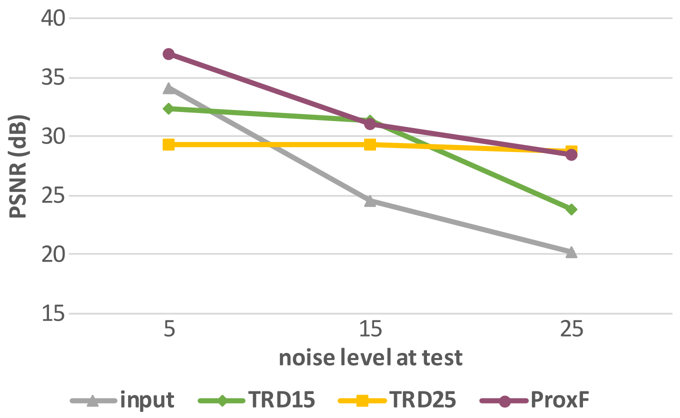
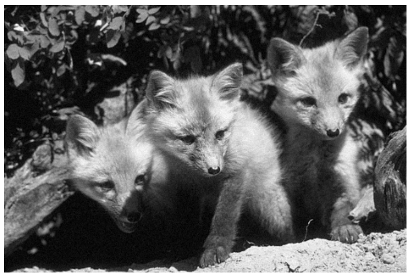
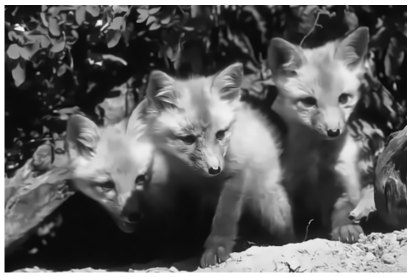


Figure 6.2: Analysis of model generality on image denoising. In this plot, "TRD15" denotes the TRD model trained at noise $\sigma = 15$, and "TRD25" at noise $\sigma = 25$. "ProxF" denotes our model trained with mixed noise levels in a single pass.

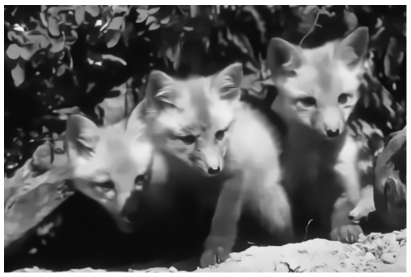
To verify the generality of our method on varying noise levels, we test our model ProxF_3^3 (trained with varying noise levels in a single pass) and two TRD models (trained at specific noise levels 15 and 25) on 3 sets of 68 images with noise $\sigma = 5, 15, 25$ respectively. The average PSNR values are reported in Fig. 6.2 and example images are shown in Fig.6.3-6.8. Despite performing slightly below the TRD models trained for the exact noise level used at test time, our method is more generic and works robustly for various noise levels. Note that our model contains 40% fewer trainable parameters than the compared TRD models. Besides,



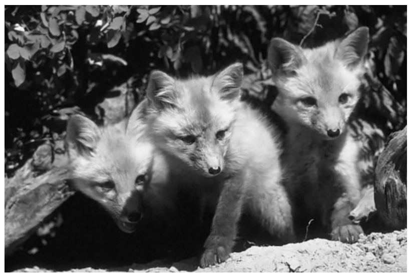
(a) Input w/ noise σ 5 (34.15dB)



(b) TRD15 (32.57dB)

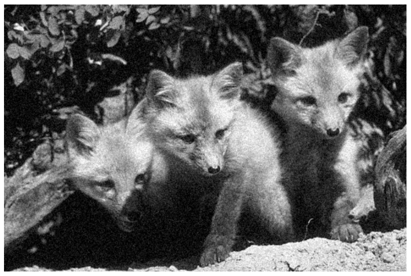


(c) TRD25 (29.33dB)

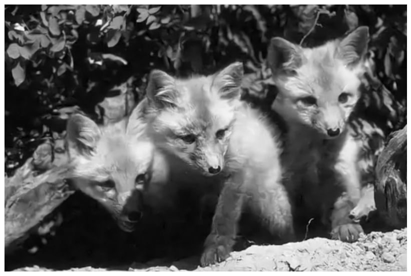


(d) ProxF (37.14dB)

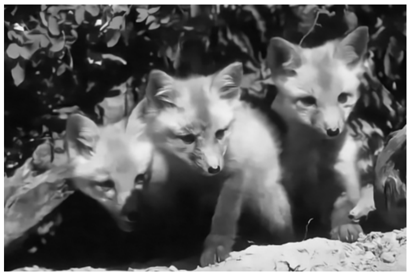
Figure 6.3: Results with input noise level σ=5.



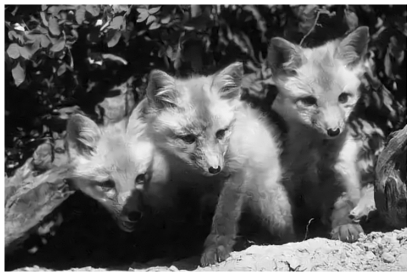
(a) Input w/ noise σ 15 (24.61dB)



(b) TRD15 (31.09dB)

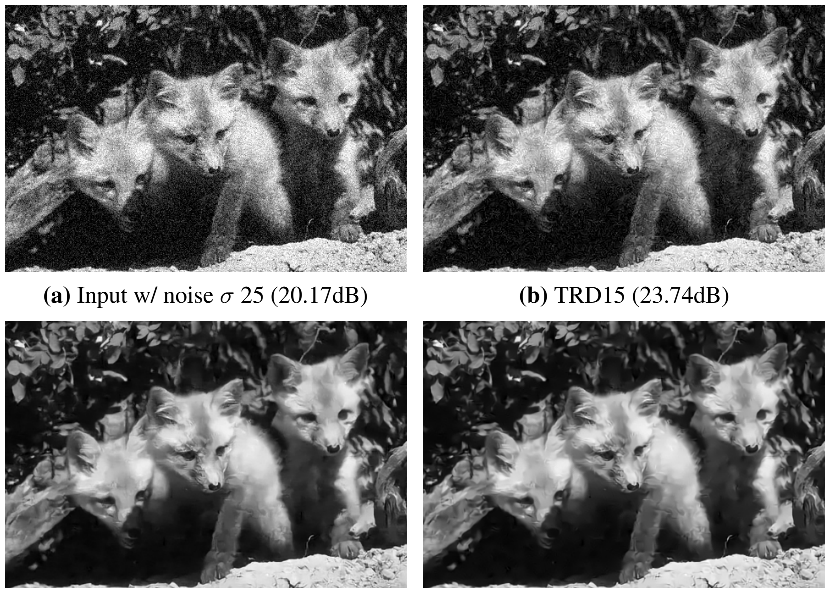


(c) TRD25 (29.31dB)



(d) ProxF (31.10dB)

Figure 6.4: Results with input noise level σ=15.


(a) Input w/ noise σ 25 (20.17dB) **(b)** TRD15 (23.74dB)

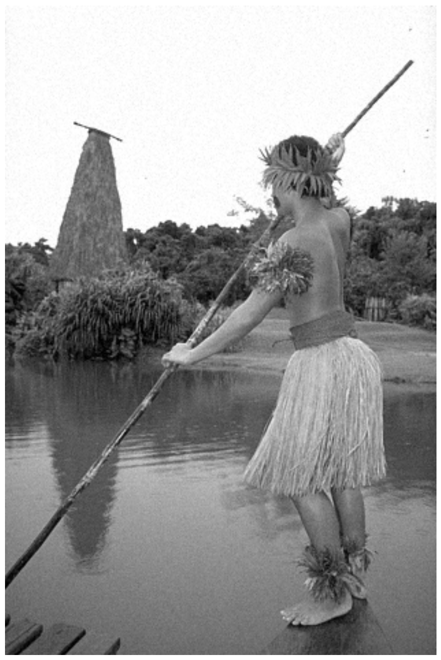
(c) TRD25 (28.44dB) **(d)** ProxF (28.45dB)

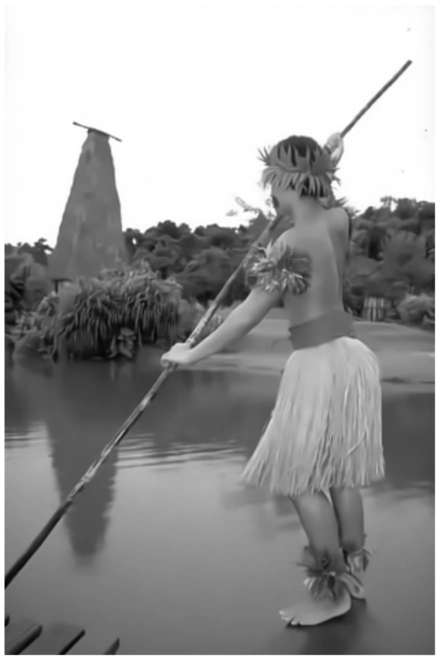
Figure 6.5: Results with input noise level σ=25.

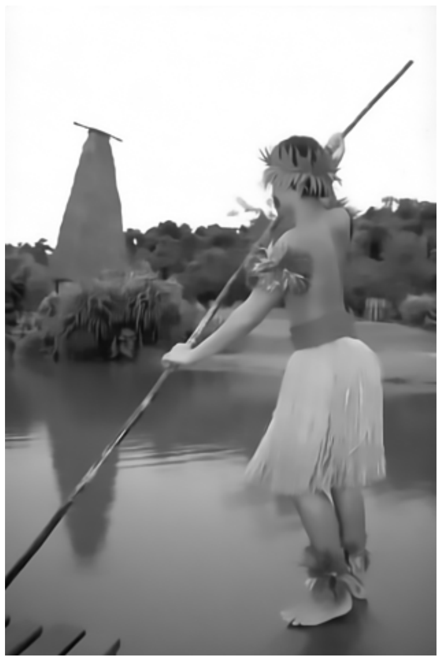
our method uses the fidelity weight λ that is learned for each noise level at training, although adjusting its value at test time can expectedly improve our results. The learned value $\lambda = 20.706$ for $\sigma = 5$, $\lambda = 2.475$ for $\sigma = 15$, and $\lambda = 0.033$ for $\sigma = 25$. In sharp contrast to discriminative methods, which are inherently specialized for a given problem setting, i.e., noise level, the proposed approach transfers across different problem settings. We demonstrate this generality below for a variety of different image reconstruction tasks.

Run-time comparison

In Table 6.2 we compare the run-time of our method and state-of-the-art methods. The experiments were performed on a laptop computer with Intel i7-4720HQ CPU and 16GB RAM. WNNM and EPLL ran out-of-memory for images over 4 megapixels in our experiments. CSF_5, TRD_5 and our method ProxF_3^3 all use "parfor" setting in Matlab. ProxF_3^3 is significantly faster than all compared generic methods (WNNM, EPLL, BM3D) and even the discriminative method CSF_5. Run-time of ProxF_3^3 is about 1.5 times that of TRD_5, which is expected as they use 9


(a) Input w/ noise σ 5 (34.15dB)


(b) TRD15 (33.29dB)


(c) TRD25 (29.95dB)

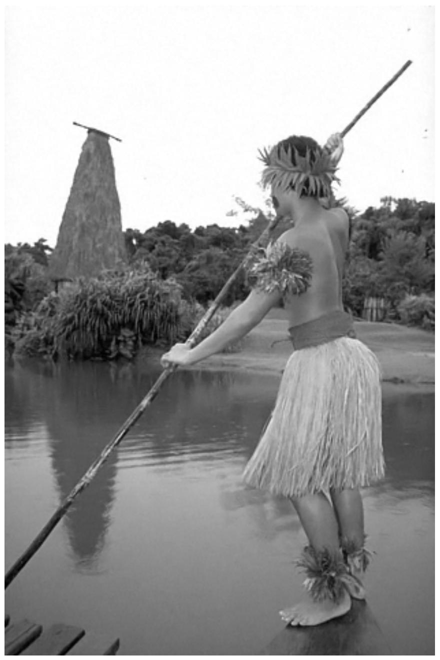
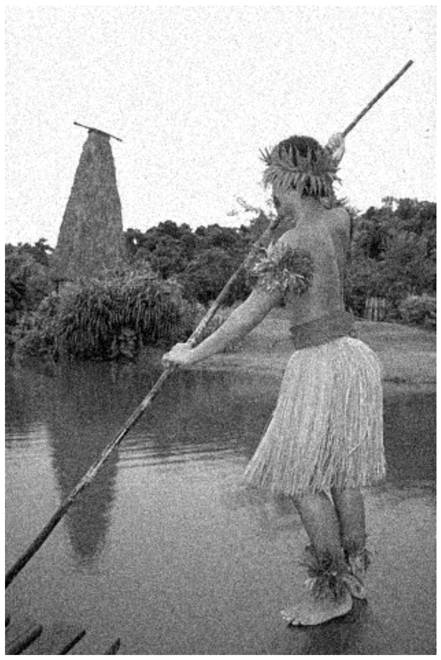
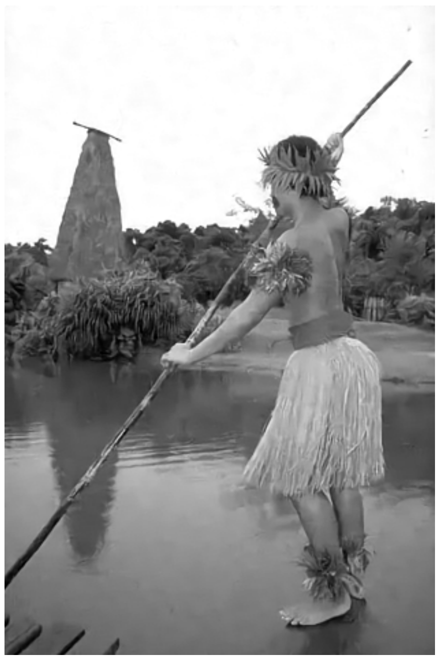

(d) ProxF (38.02dB)

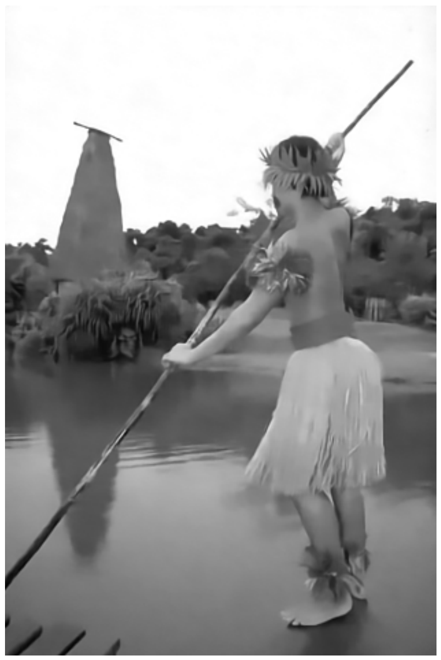
Figure 6.6: Results with input noise level σ=5.



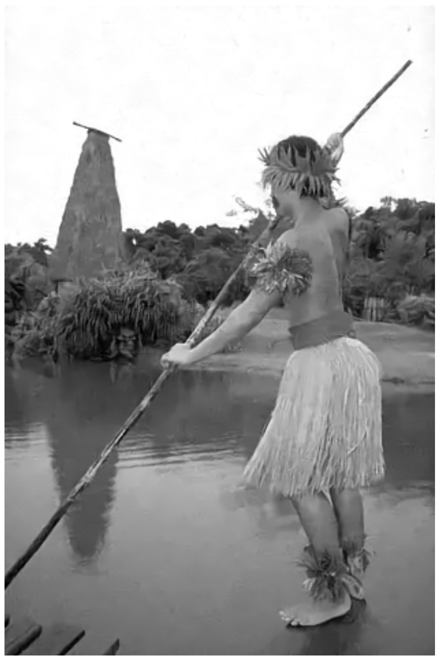
(a) Input w/ noise σ 15 (24.61dB)



(b) TRD15 (32.31dB)

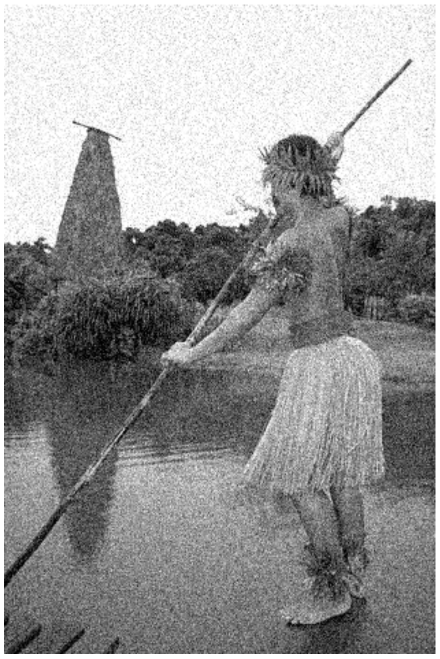


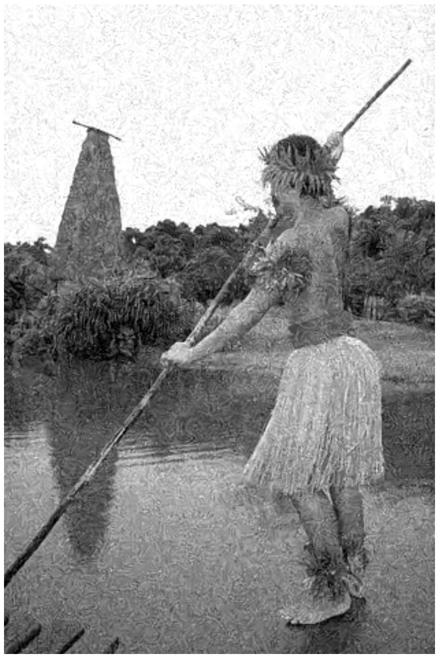
(c) TRD25 (30.11dB)

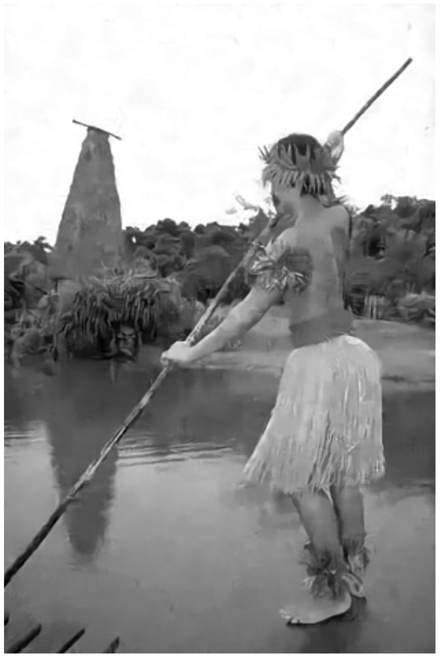


(d) ProxF (32.03dB)

Figure 6.7: Results with input noise level σ=15.


(a) Input w/ noise σ 25 (20.17dB)


(b) TRD15 (23.94dB)


(c) TRD25 (29.72dB)

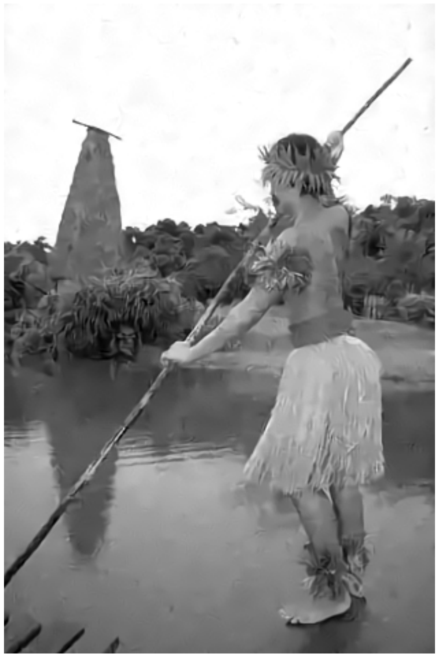

(d) ProxF (29.40dB)

Figure 6.8: Results with input noise level σ=25.

Table 6.2: Run-time (seconds) comparison for image denoising on different size images.

Image size	256^2	512^2	1024^2	2048^2	4096^2
WNNM	157.73	657.75	2759.79	-	-
EPLL	29.21	111.52	463.71	-	-
BM3D	0.78	3.45	15.24	62.81	275.39
CSF_5	1.23	2.22	7.35	27.08	93.66
TRD_5	0.39	0.71	2.01	7.57	29.09
ProxF_3^3	0.60	1.19	3.45	12.97	56.19
ProxF_3^3 (Halide)	0.11	0.26	1.60	5.61	20.85

versus 5 diffusion steps in total. In addition, we implement our method in Halide language [82], which has become popular recently for high-performance image processing applications, and report the run-time on the same CPU as mentioned above.

Deconvolution

Our general model supports image deconvolution tasks in the HQS framework. In this experiment, we train a model with an ensemble of denoising and deconvolution tasks on 400 images (100×100 pixels) cropped from [87], in which 250 images are generated for denoising tasks with random noise levels σ varying between 5 and 25, and the other 150 images are generated by blurring the images with random 25×25 kernels (PSFs) and then adding Gaussian noise with σ between 1 and 5. All input images are quantized to 8 bits.

We compare our method with state-of-the-art non-blind deconvolution methods including Levin *et al* [68], Schmidt *et al* [93] and CSF [92] on the benchmark dataset of [63]. Note that TRD [18] does not support non-blind deconvolution. We test the methods on 32 images from [64] and report the average PSNR values in Table 6.3. The results of compared methods are quoted from [92]. We run a grid search on the adjustable fidelity weight λ (the same value for all the 32 test images) and report the best result in Table 6.3. In Fig. 6.9, we show our results with different λ. Within a fairly wide range of λ, our method outperforms all previous methods. In Fig.6.10, we show example images of our results and more images

Table 6.3: Average PSNR (dB) on 32 images from [64] for non-blind deconvolution.

Input	Levin [68]	Schmidt [93]	CSF_3^{pw}	ProxF_3^3
22.86	32.73	33.97	33.48	34.34

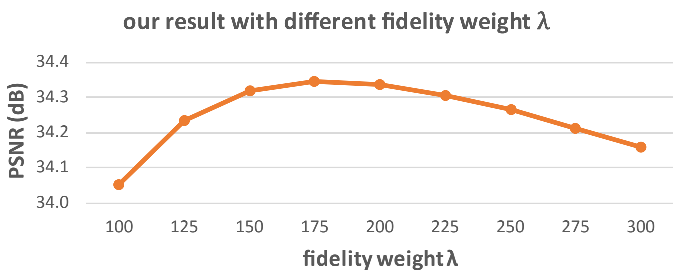
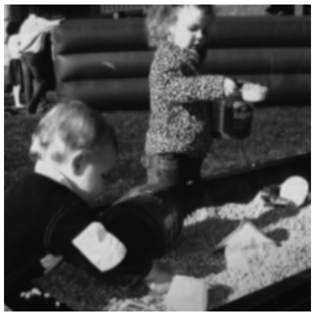
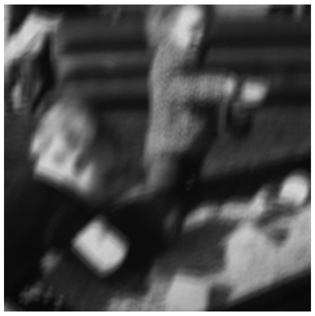


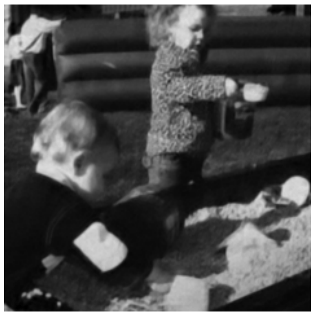
Figure 6.9: Our results with different fidelity weight λ for the non-blind deconvolution experiment reported in Table 6.3.

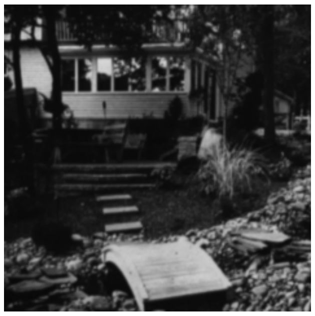
can be found in the supplementary material of [109].

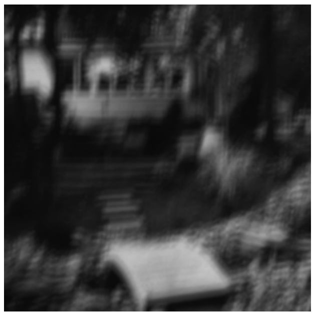
In addition, in Fig.6.11 -6.12, we show a comparison for non-blind deconvolution on the dataset from Schuler *et al* [94] that features Gaussian blur with standard deviation 1.6 and noise level $\sigma = 2$. We compare our method ProxF_3^3 with EPLL [118], Krishnan *et al* [58], Levin *et al* [67], DEB-BM3D [24], IDD-BM3D [24] and MLP [94]. Note that the MLP method is tailored for the task of non-blind deconvolution only and is trained with exactly the same (single) blur kernel and noise level as those at test time [94]. In contrast, our method ProxF_3^3 is trained with an ensemble of denoising and deconvolution tasks including various blur kernels and noise levels. In particular, our training data does not comprise any Gaussian blur kernels. Nonetheless, our method is close to MLP in terms of PSNR while at the same time outperforming all other competitors. Please zoom in the figures for better view.

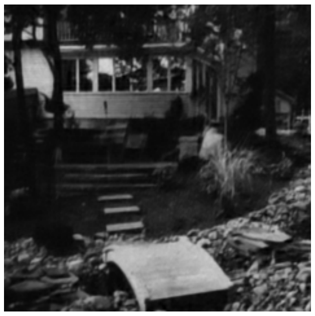

(a) Ground truth

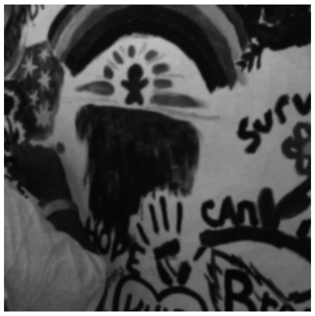

(b) Input (23.69dB)

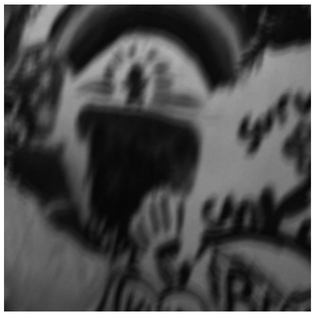

(c) ProxF$_3^3$ (33.81dB)

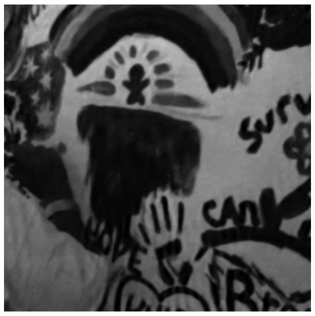

(d) Ground truth

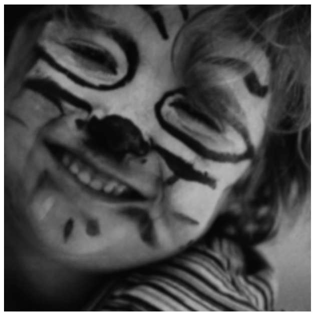

(e) Input (22.61dB)

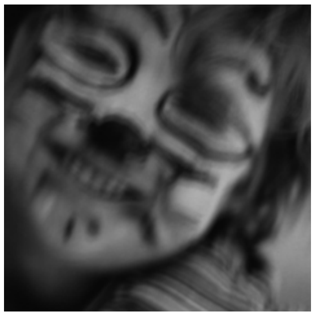

(f) ProxF$_3^3$ (33.24dB)


(g) Ground truth


(h) Input (23.74dB)


(i) ProxF$_3^3$ (33.86dB)


(j) Ground truth


(k) Input (24.58dB)

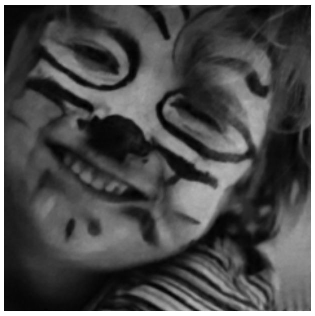

(l) ProxF$_3^3$ (34.74dB)

Figure 6.10: Results on images with blur kernel #1 in the dataset.

(a) Ground truth

(b) Input (32.98dB)

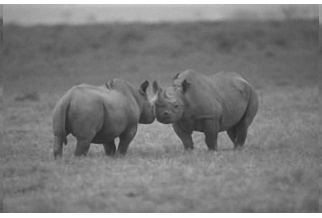
(c) EPLL (35.25dB)

(d) Krishnan (35.12dB)

(e) Levin (35.00dB)

(f) DEB-BM3D (35.01dB)

(g) IDD-BM3D (35.26dB)


(h) MLP (35.36dB)

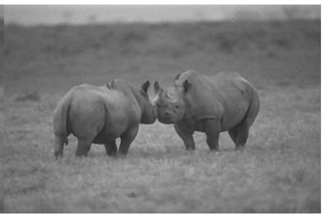
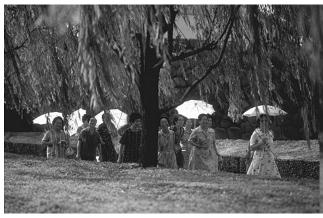
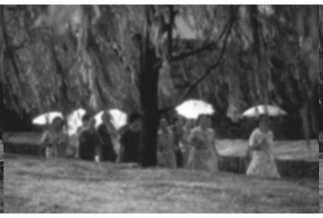
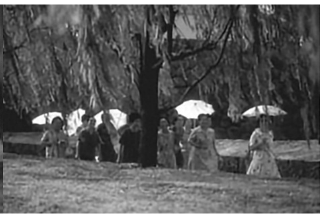
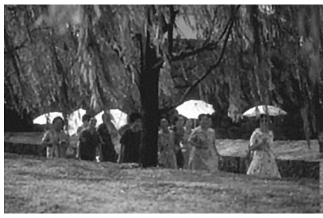

(i) ProxF$_3^3$ (35.39dB)

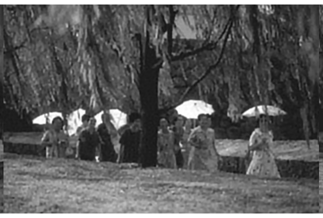
Figure 6.11: Results on non-blind deconvolution with Gaussian blur.

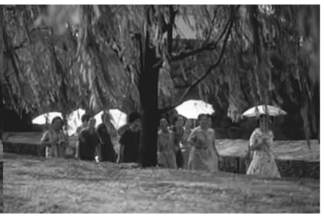

(a) Ground truth

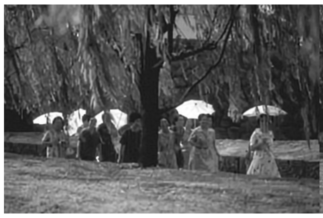

(b) Input(23.26dB)

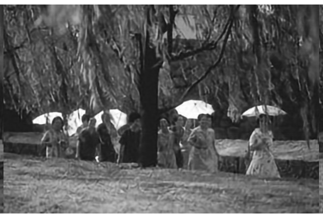

(c) EPLL (25.48dB)


(d) Krishnan (25.56dB)


(e) Levin (25.53dB)


(f) DEB-BM3D (25.58dB)


(g) IDD-BM3D (25.73dB)


(h) MLP (25.93dB)

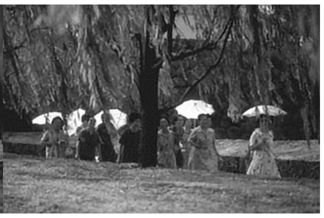

(i) ProxF$_3^3$ (25.76dB)

Figure 6.12: Results on non-blind deconvolution with Gaussian blur.

Collaborative with existing priors

As shown above, even though the fidelity weight λ is trainable, our method allows users to adjust its value for better performance at test time. Moreover, this property makes it possible to combine our model (after being trained) with existing state-of-the-art priors at test time, in which case λ typically needs to be adjusted. Again, this is not possible with previous discriminative methods (CSF, TRD).

In Fig. 6.13 we show an example to incorporate a non-local patch similarity prior (BM3D [21]) with our method to further improve the denoising quality. BM3D performs well in removing noise especially in smooth regions but usually over-smoothes edges and textures. Our original model (ProxF$_3^5$) well preserves sharp edges however sometimes introduces artifacts in smooth regions when the input noise level is high. By combining those two methods, which is easy with our HQS framework, the result is improved both visually and quantitatively.

We give the derivation of the proposed hybrid method below. Let $\mathcal{S}(\mathbf{x})$ represents the non-local patch similarity prior. The objective function is:

$$\frac{\lambda}{2}||\mathbf{b}-\mathbf{A}\mathbf{x}||_2^2+\sum_{i=1}^{N}\phi_i(\mathbf{F}_i\mathbf{x})+\tau\mathcal{S}(\mathbf{x}) \tag{6.12}$$

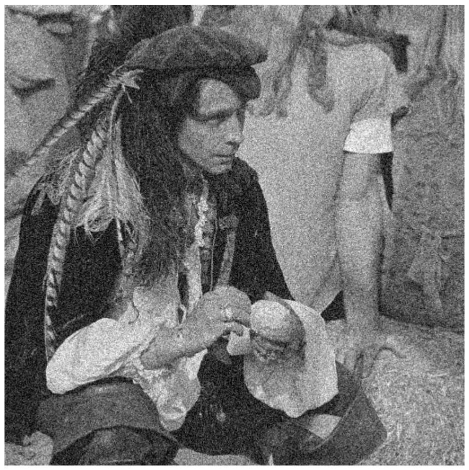
Applying the HQS technique described in Section 6.2, we relax the objective to be:

$$\begin{aligned}\frac{\lambda}{2}||\mathbf{b}-\mathbf{A}\mathbf{x}||_2^2+\frac{\rho}{2}||\mathbf{z}-\mathbf{x}||_2^2+\sum_{i=1}^{N}\phi_i(\mathbf{F}_i\mathbf{z})\\ +\frac{\rho_s}{2}||\mathbf{v}-\mathbf{x}||_2^2+\tau\mathcal{S}(\mathbf{v})\end{aligned} \tag{6.13}$$

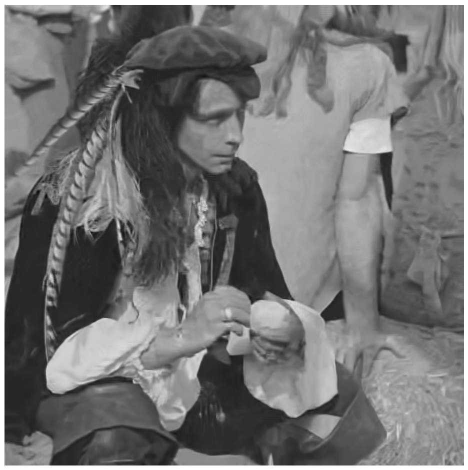
Then we minimize Eq. 6.13 by alternately solving the following 3 subproblems:

$$\begin{aligned}\mathbf{z}^t &= \mathbf{prox}_\Theta(\mathbf{x}^{t-1})\\ \mathbf{v}^t &= \operatorname*{argmin}_{\mathbf{v}}\frac{\rho_s^t}{2}||\mathbf{v}-\mathbf{x}^{t-1}||_2^2+\tau\mathcal{S}(\mathbf{v})\approx \text{BM3D}\ (\mathbf{x}^{t-1},\frac{\tau}{\rho_s^t})\\ \mathbf{x}^t &= \operatorname*{argmin}_{\mathbf{x}}\lambda||\mathbf{b}-\mathbf{A}\mathbf{x}||_2^2+\rho^t||\mathbf{z}^t-\mathbf{x}||_2^2+\rho_s^t||\mathbf{v}^t-\mathbf{x}||_2^2,\end{aligned} \tag{6.14}$$

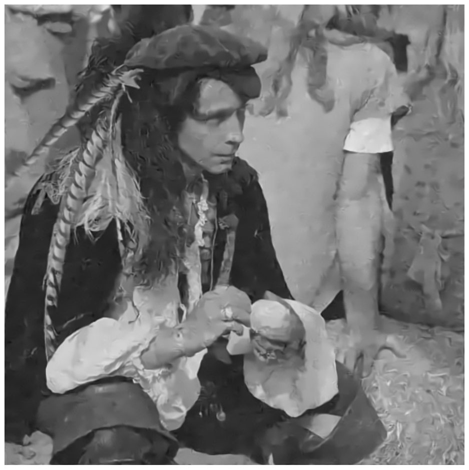
where $\mathbf{prox}_\Theta$ is from our previous training, and the $\mathbf{v}^t$ subproblem is *approxi-*



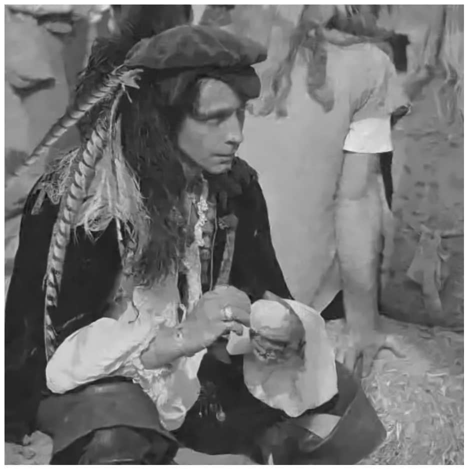
(a) Input (20.17dB)



(b) BM3D (29.62dB)

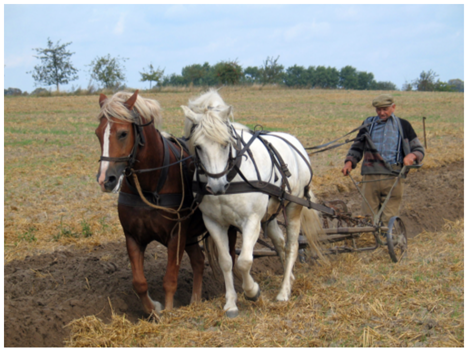


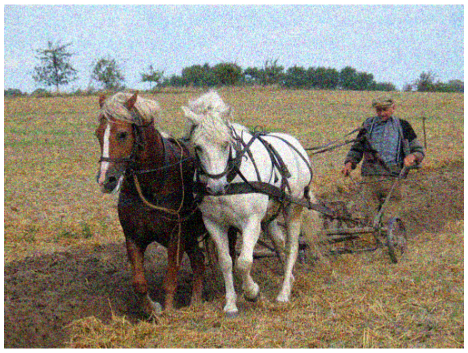
(c) ProxF$_3^5$ (29.48dB)

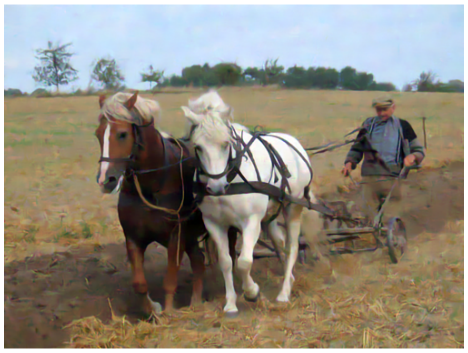


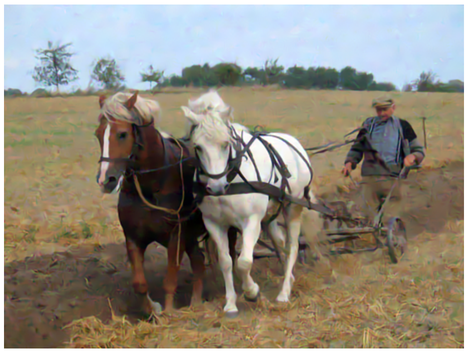
(d) ProxF$_3^5$ + BM3D (29.74dB)

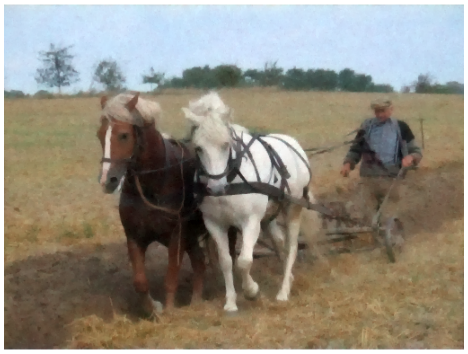
Figure 6.13: Experiment on incorporating non-local patch similarity prior (BM3D) with our model after being trained. The input noise level $\sigma = 25$. Please zoom in for better view.


(a) Ground truth


(b) Input (20.18dB)


(c) TRD$_5$ (28.06dB)


(d) ProxF$_3^5$ (27.80dB)


(e) TV + cross (26.89dB)

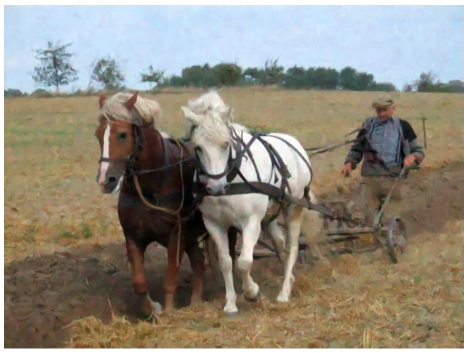

(f) ProxF$_3^5$ + cross (28.69dB)

Figure 6.14: Experiment on incorporating a color prior [45] with our model after being trained. The input noise level $\sigma = 25$. (e,f) show the results by combining TV denoising with a cross-channel prior, and our method with cross-channel prior, respectively. Please zoom in for better view.

mated by running BM3D software on $\mathbf{x}^{t-1}$ with noise parameter $\frac{\tau}{\rho_s^t}$ following [45].

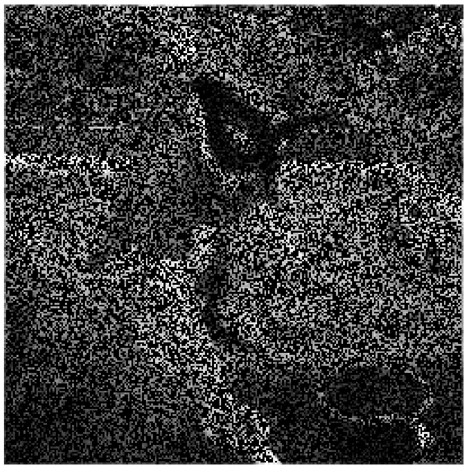
Similarly, our method can incorporate color image priors (e.g., cross-channel edge-concurrence prior [45]) to improve test results on color images, despite our model being trained on gray-scale images. An example is shown in Fig. 6.14. The hybrid method shares the advantages of our original model that effectively preserves edges and textures and the cross-channel prior that reduces color artifacts.

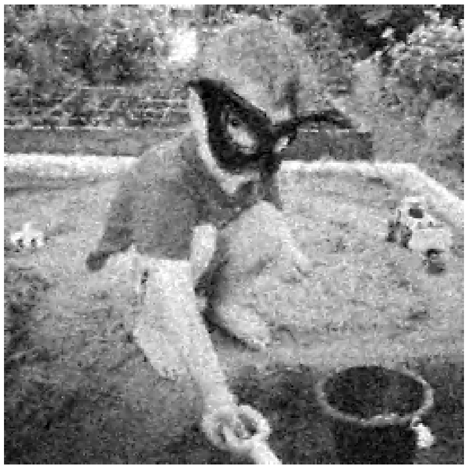
Transferability to unseen tasks

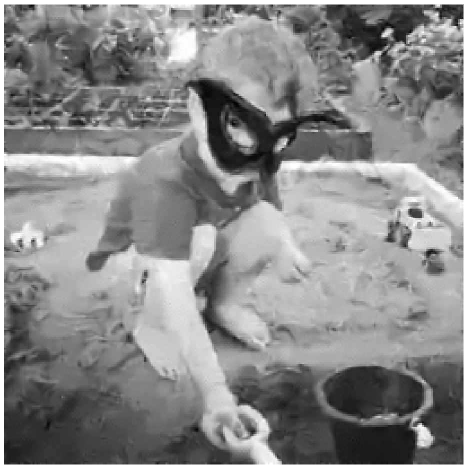
Our method allows for new data-fidelity terms that are not contained in training, with no need for re-training. We demonstrate this flexibility with an experiment on the joint denoising and inpainting task shown in Fig. 6.15. In this experiment, 60% pixels of the input image are missing, and the measured 40% pixels are corrupted with Gaussian noise with $\sigma = 15$. Let vector $\mathbf{a}$ be the binary mask for measured pixels. The sensing matrix $\mathbf{A}$ in Eq. 6.1, assumed to be known, is a binary diagonal matrix (hence $\mathbf{A} = \mathbf{A}^\mathsf{T} = \mathbf{A}^\mathsf{T}\mathbf{A}$) with diagonal elements $\mathbf{a}$. To reuse our model trained on denoising/deconvolution tasks, we only need to specify $\mathbf{A}$ and λ. The subproblems of our HQS framework are given in Eq. 6.15.

$$\begin{aligned}\mathbf{z}^t &= \mathbf{prox}_\Theta(\mathbf{x}^{t-1}),\\ \mathbf{x}^t &= (\lambda\mathbf{A}^\mathsf{T}\mathbf{b} + \rho^t\mathbf{z}^t)./(\lambda\mathbf{a} + \rho^t),\end{aligned} \tag{6.15}$$

where ./ indicates element-wise division.


(a) Input


(b) Delaunay interp.(23.19dB)


(c) ProxF$^5_3$ (25.10dB)

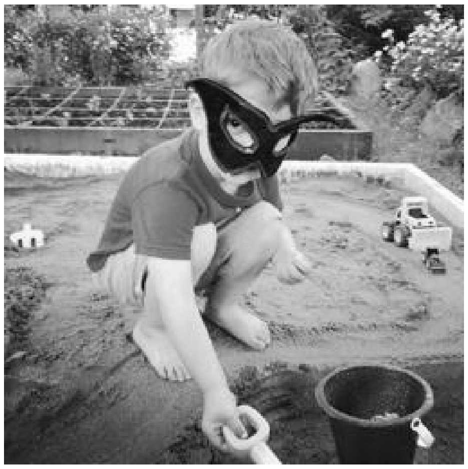
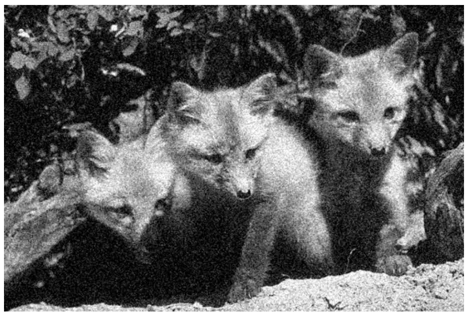
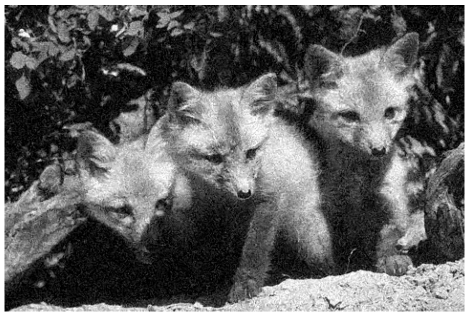
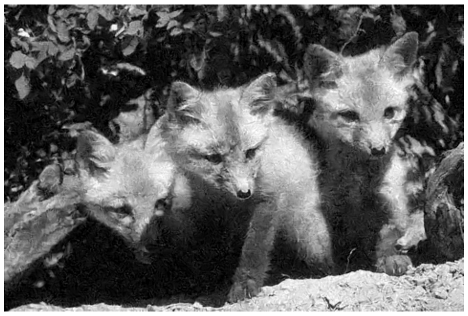

(d) Ground truth

Figure 6.15: Experiment on joint denoising and inpainting task. The input image (a) misses 60% pixels, and is corrupted with noise $\sigma = 15$. Our method takes the result of Delaunary interpolation (b) as the initial estimation $\mathbf{x}^0$. Please zoom in for better view.

(a) Ground truth


(b) Noisy input (20.18dB)


(c) Iter 1 (22.85dB)


(d) Iter 2 (25.93dB)

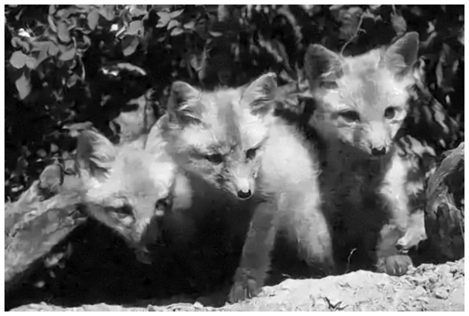
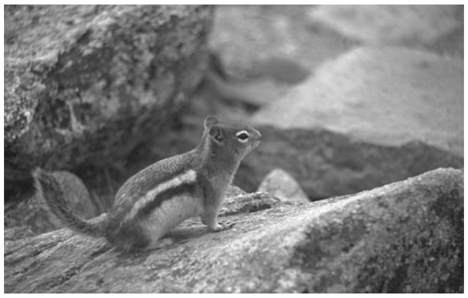
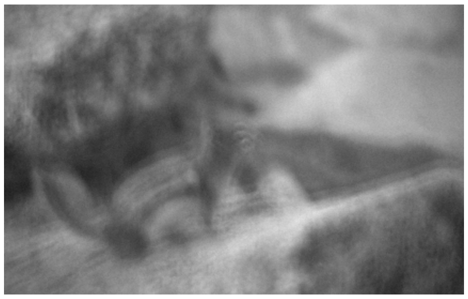
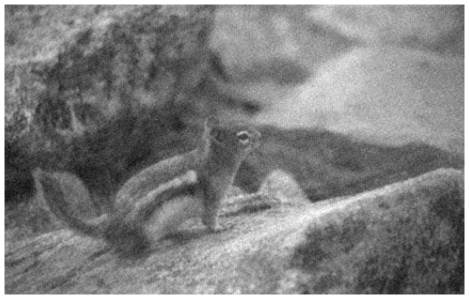
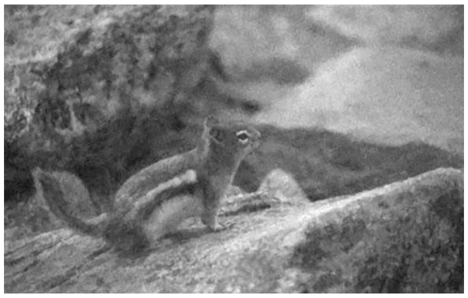

(e) Iter 3 (28.14dB)

Figure 6.16: Results at each HQS iteration of our method on image denoising with noise level $\sigma = 25$. Inside brackets show the PSNR values. Please zoom in for better view.


(a) Ground truth


(b) Blurry input (23.37dB)


(c) Iter 1 (27.32dB)


(d) Iter 2 (28.48dB)

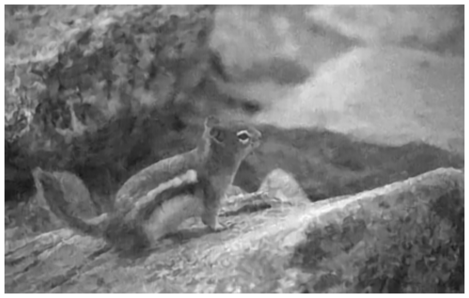

(e) Iter 3 (29.36dB)

Figure 6.17: Results at each HQS iteration of our method on non-blind deconvolution with a 25×25 PSF and noise level $\sigma = 3$. Inside brackets show the PSNR values. Please zoom in for better view.

Table 6.4: Test with different HQS iterations (T) and model stages (K) for image denoising. Average PSNR (dB) results on 68 images from [64] with noise $\sigma = 15$ and 25 are reported (before and after "/" in each cell respectively).

		# HQS iterations		
		1	3	5
# stages	1	29.80 / 26.81	30.89 / 28.12	30.96 / 28.28
	3	30.54 / 27.82	30.91 / 28.19	31.00 / 28.42
	5	30.54 / 27.83	30.92 / 28.18	-

Analysis of convergence and model complexity

To better understand the convergence of our method, in Fig. 6.16 and 6.17 we show the results of each HQS iteration of our method on denoising and non-blind deconvolution.

To understand the effect of model complexity and the number of HQS iteration on results, in Table 6.4 we report test results of our method using models trained with different HQS iterations (T in Algorithm 8), and with different stages in $\mathbf{prox}_\Theta$ (K in Algorithm 8). The results are generated using the λ values learned at training without post-tuning.

Connections to plug-and-play priors

We noticed that recent plug-and-play methods [72, 85, 103] make a similar observation that the $\mathbf{z}^t$-update step in 6.3 can be interpreted as a Gaussian denoising processing on $\mathbf{x}^{t-1}$. However, while they replace the $\mathbf{z}^t$-update step with existing generic Gaussian denoisers, our method trains such step as part of the whole proximal optimization iterations by discriminative learning for good trade-off between high-quality results and time-efficiency.

6.4 Conclusion and future work

In this paper, we proposed the trainable proximal fields model, a generalization of traditional proximal operators. By combining advanced proximal optimization algorithms and discriminative learning techniques, a single training pass leads to a transferable model useful for a variety of image restoration tasks and problem conditions. Furthermore, our method is flexible and can be combined with existing priors and likelihood terms after being trained, allowing to improve image quality on a task at hand. In spite of this generality, our method achieves comparable run-time efficiency as previous discriminative approaches, making it suitable for high-resolution image restoration and mobile vision applications.

We believe that in future work, our framework incorporating advanced optimization with discriminative learning techniques can be extended to deep learning, for training more compact and shareable models, and to solve high-level vision problems. Another plan is to train our models for ensemble tasks with larger datasets and use advanced learning optimization techniques, which can potentially further improve results.

Chapter 7

Conclusions

As a classic topic that has been studied for decades, image restoration is still a very active research area. While significant progress has been made, developing more time-, memory- and power-efficient methods are still highly desirable especially as the rise of mobile imaging systems such as smartphone cameras, ToF cameras, autonomous vehicles, etc. Recent advances in machine learning and data-driven techniques have been inspiring new perspectives and strategies for image restoration problems. This thesis presents our researches on these topics and addresses several image restoration problems for applications in computational imaging including ToF imaging and digital photography.

While continuous-wave ToF cameras have shown great promise at low-cost depth imaging, they suffer from limited depth of field and low spatial resolution. In Chapter 3, we develop a computational method to remove the lens blur and increase image resolution of off-the-shelf ToF cameras. In contrast to previous work, our method solves the depth and amplitude image directly from the raw sensor data and supports more advanced ToF cameras that use multiple frequencies, phases and exposures.

Taking personal photographs have become increasingly common due to the popularity of mobile cameras. However, photos taken by hand-held cameras are likely to suffer from blur caused by camera shake during exposure. Therefore, removing such blur and recovering sharp images as a post-process is highly desirable. In Chapter 4 we develop a blind image deblurring method that is purely based on simple stochastic random-walk sampling for optimization, which allows us to easily test new priors for both image and blur kernel. We demonstrate that this simple framework in combination with different priors produces comparable results to the much more complex state-of-the-art deblurring algorithms.

Blur causes even more serious issues for text document photographs, as slight blur can prevent existing OCR techniques from extracting correct text from the images. In Chapter 5 we address the blind deblurring problem specifically for common text document photographs. Observing that the latter are mostly composed of high-order structures, our method proposes to capture such domain property by a series of iteration- and scale-wise high-order filters as well as customized response functions. These filters and functions are learned from data by discriminative learning approach and form an end-to-end network that can efficiently estimate both blur kernel and sharp/legible text images.

Discriminative learning approaches have been recently proposed for effective image restoration, achieving convincing trade-off between image quality and computational efficiency. However, these methods require separate training for each restoration task and problem condition, making it time-consuming and difficult to encompass all tasks and conditions during training. In Chapter 6 we combine the discriminative learning idea and formal numerical optimization method, to learn image priors that require a single-pass training and share across various tasks and conditions, while keeping the efficiency as previous discriminative methods. Moreover, after being trained, our model can be easily combined with other likelihood or priors to address unseen restoration tasks or further improve the image quality.

Throughout this thesis work, developing both effective image priors and efficient optimization solvers have been the central tasks for a given image restoration problem. In Chapter 3 we investigate the classic prior (TGV) and the advanced numerical optimization methods (ADMM and Levenberg-Marquardt algorithm). In Chapter 4 we develop a stochastic-sampling based optimization method for solving general image priors. In Chapter 5, we apply discriminative learning approach to learn the priors *and* the iterative optimization process from data. The learned models are highly optimized and customized for the trained task. In Chapter 6 we combine the numerical optimization method (HQS) with the discriminative learning approach to learn prior models that can be shared across a variety of restoration tasks while keeping the efficiency as previous discriminative learning methods. While this is a first attempt, I believe that the emerging between formal numerical optimization techniques and the data-driven machine learning techniques will stimulate more new ideas and successes in the field of image restoration, and I would like to keep investigating on it as future work.

Bibliography

[1] Robust deblurring software. www.cse.cuhk.edu.hk/~leojia/deblurring.htm. → pages xii, 59, 68, 69, 70, 71, 75, 86, 87, 89, 91

[2] B. C. A. Buades and J. M. Morel. A review of image denoising algorithms, with a new one. *Multiscale Modeling and Simulation*, 4(2):490–530, 2005. → pages 1, 9

[3] S. Agarwal, K. Mierle, and Others. Ceres solver. http://ceres-solver.org. → pages 38

[4] F. Anscombe. The transformation of poisson, binomial and negative-binomial data. *Biometrika*, 35(3/4):246–254, 1948. → pages 65

[5] S. Anwar, C. Phuoc Huynh, and F. Porikli. Class-specific image deblurring. In *ICCV 2015*. → pages 14

[6] H. Avron, A. Sharf, C. Greif, and D. Cohen-Or. 1-sparse reconstruction of sharp point set surfaces. *ACM Transactions on Graphics*, 29(5):135, 2010. → pages 25

[7] J. Balzer and S. Soatto. Second-order shape optimization for geometric inverse problems in vision. *arXiv preprint arXiv:1311.2626*, 2013. → pages 25

[8] A. Barbu. Training an active random field for real-time image denoising. *IEEE Transactions on Image Processing*, 18(11):2451–2462, 2009. → pages 101

[9] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011. → pages 21, 27, 28

[10] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011. → pages 6, 96

[11] K. Bredies, K. Kunisch, and T. Pock. Total generalized variation. *SIAM Journal on Imaging Sciences*, 3(3):492–526, 2010. → pages 25

[12] H. Bristow, A. Eriksson, and S. Lucey. Fast convolutional sparse coding. In *CVPR 2013*. → pages 10

[13] H. C. Burger, C. J. Schuler, and S. Harmeling. Image denoising: Can plain neural networks compete with BM3D? In *CVPR 2012*. → pages 2, 12

[14] G. Casella and C. P. Robert. Monte carlo statistical methods, 1999. → pages 46

[15] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011. → pages 2, 96

[16] X. Chen, X. He, J. Yang, and Q. Wu. An effective document image deblurring algorithm. In *CVPR 2011*, . → pages 5, 15

[17] Y. Chen, T. Pock, R. Ranftl, and H. Bischof. Revisiting loss-specific training of filter-based mrfs for image restoration. In *German Conference on Pattern Recognition 2013*, . → pages 101

[18] Y. Chen, W. Yu, and T. Pock. On learning optimized reaction diffusion processes for effective image restoration. In *CVPR 2015*, . → pages 2, 5, 12, 95, 97, 98, 101, 108

[19] H. Cho, J. Wang, and S. Lee. Text image deblurring using text-specific properties. In *ECCV 2012*. → pages 5, 15

[20] S. Cho and S. Lee. Fast motion deblurring. In *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, pages 145:1–145:8, 2009. → pages xii, 14, 54, 59, 60, 61, 68, 70, 71, 75, 79

[21] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image restoration by sparse 3d transform-domain collaborative filtering. In *Electronic Imaging 2008*. → pages 13, 113

[22] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, 2007. $\rightarrow$ pages 1, 9, 101

[23] A. Danielyan, V. Katkovnik, and K. Egiazarian. Image deblurring by augmented lagrangian with bm3d frame prior. In *Proc. WITMSE*, pages 16–18, 2010. $\rightarrow$ pages 13

[24] A. Danielyan, V. Katkovnik, and K. Egiazarian. Bm3d frames and variational image deblurring. *IEEE Transactions on Image Processing*, 21 (4):1715–1728, 2012. $\rightarrow$ pages 13, 109

[25] W. Dong, L. Zhang, G. Shi, and X. Li. Nonlocally centralized sparse representation for image restoration. *IEEE Transactions on Image Processing*, 22(4):1620–1630, 2013. $\rightarrow$ pages 1, 9, 10

[26] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, 2006. $\rightarrow$ pages 2, 10, 101

[27] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single photograph. In *ACM Transactions on Graphics (Proc. SIGGRAPH)*, pages 787–794, 2006. $\rightarrow$ pages 12, 13, 44, 49, 50, 59, 60, 68, 70

[28] D. Ferstl, C. Reinbacher, R. Ranftl, M. Ruether, and H. Bischof. Image guided depth upsampling using anisotropic total generalized variation. In *ICCV 2013*. $\rightarrow$ pages 17, 25

[29] D. Freedman, Y. Smolin, E. Krupka, I. Leichter, and M. Schmidt. SRA: Fast removal of general multipath for ToF sensors. In *ECCV 2014*. $\rightarrow$ pages 18

[30] D. Geman and C. Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE Transactions on Image Processing*, 4(7):932–946, 1995. $\rightarrow$ pages 2, 6, 19, 78, 96, 140

[31] J. P. Godbaz, M. J. Cree, and A. A. Dorrington. Extending amcw lidar depth-of-field using a coded aperture. In *ACCV 2010*. $\rightarrow$ pages 17, 23, 24, 29, 39, 41

[32] J. P. Godbaz, M. J. Cree, and A. A. Dorrington. Blind deconvolution of depth-of-field limited full-field lidar data by determination of focal

parameters. In *IS&T/SPIE Electronic Imaging*, pages 75330B–75330B, 2010. → pages 17

[33] S. B. Gokturk, H. Yalcin, and C. Bamji. A time-of-flight depth sensor-system description, issues and solutions. In *CVPR Workshops 2004*. → pages 18, 23, 41

[34] A. Goldstein and R. Fattal. Blur-kernel estimation from spectral irregularities. In *ECCV 2012*. → pages 14

[35] T. Goldstein and S. Osher. The split bregman method for l1-regularized problems. *SIAM Journal on Imaging Sciences*, 2(2):323–343, 2009. → pages 2, 49

[36] J. Gregson, F. Heide, M. B. Hullin, M. Rouf, and W. Heidrich. Stochastic Deconvolution. In *CVPR 2013*. → pages 43, 45, 46, 50, 53, 63

[37] S. Gu, L. Zhang, W. Zuo, and X. Feng. Weighted nuclear norm minimization with application to image denoising. In *CVPR 2014*. → pages 1, 5, 9, 10, 101

[38] A. Gupta, N. Joshi, C. L. Zitnick, M. Cohen, and B. Curless. Single image deblurring using motion density functions. In *ECCV 2010*. → pages 14

[39] U. Hahne and M. Alexa. Exposure fusion for time-of-flight imaging. In *Computer Graphics Forum*, volume 30, pages 1887–1894, 2011. → pages 18

[40] S. Harmeling, H. Michael, and B. Schölkopf. Space-variant single-image blind deconvolution for removing camera shake. In *NIPS 2010*. → pages 14

[41] F. Heide, W. Heidrich, and G. Wetzstein. Fast and flexible convolutional sparse coding. In *CVPR 2015*. → pages 10

[42] F. Heide, M. B. Hullin, J. Gregson, and W. Heidrich. Low-budget transient imaging using photonic mixer devices. *ACM Transactions on Graphics*, 32 (4):45, 2013. → pages 23, 41

[43] F. Heide, M. Rouf, M. B. Hullin, B. Labitzke, W. Heidrich, and A. Kolb. High-quality computational imaging through simple lenses. *ACM Transactions on Graphics*, 32(5), September 2013. → pages 61, 62

[44] F. Heide, M. Rouf, M. B. Hullin, B. Labitzke, W. Heidrich, and A. Kolb. High-quality computational imaging through simple lenses. *ACM Transactions on Graphics*, 32(5):149, 2013. → pages 29

[45] F. Heide, M. Steinberger, Y.-T. Tsai, M. Rouf, D. Pajak, D. Reddy, O. Gallo, J. Liu, W. Heidrich, K. Egiazarian, et al. Flexisp: a flexible camera image processing framework. *ACM Transactions on Graphics*, 33 (6):231, 2014. → pages 115, 116

[46] M. Hirsch, C. J. Schuler, S. Harmeling, and B. Scholkopf. Fast removal of non-uniform camera shake. In *ICCV 2011*, . → pages 14, 60, 68, 70

[47] M. Hirsch, S. Sra, B. Scholkopf, and S. Harmeling. Efficient filter flow for space-variant multiframe blind deconvolution. In *CVPR 2010*, . → pages 92, 93

[48] M. Hradiš, J. Kotera, P. Zemcík, and F. Šroubek. Convolutional neural networks for direct text deblurring. In *BMVC 2015*. → pages 5, 15, 77, 86, 87, 88, 89, 90, 91, 92, 93, 94

[49] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *UIST 2011*. → pages 17

[50] V. Jain and H. Seung. Natural image denoising with convolutional networks. *NIPS 2009*. → pages 2, 12

[51] J. Jancsary, S. Nowozin, T. Sharp, and C. Rother. Regression tree fields - an efficient, non-parametric approach to image labeling problems. In *CVPR 2012*. → pages 2, 12

[52] N. Joshi, S. B. Kang, C. L. Zitnick, and R. Szeliski. Image deblurring using inertial measurement sensors. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 29(4):30, 2010. → pages 43

[53] M. Kiechle, S. Hawe, and M. Kleinsteuber. A joint intensity and depth co-sparse analysis model for depth map super-resolution. In *ICCV 2013*. → pages 17

[54] S. Kindermann, S. Osher, and P. W. Jones. Deblurring and denoising of images by nonlocal functionals. *Multiscale Modeling & Simulation*, 4(4): 1091–1115, 2005. → pages 13

[55] A. Kirmani, A. Benedetti, and P. A. Chou. Spumic: Simultaneous phase unwrapping and multipath interference cancellation in time-of-flight cameras using spectral methods. In *ICME 2013*. → pages 18

[56] F. Knoll, K. Bredies, T. Pock, and R. Stollberger. Second order total generalized variation (tgv) for mri. *Magnetic resonance in medicine*, 65(2): 480–491, 2011. → pages 25

[57] R. Köhler, M. Hirsch, B. Mohler, B. Schölkopf, and S. Harmeling. Recording and playback of camera shake: benchmarking blind deconvolution with a real-world database. In *ECCV 2012*. → pages x, xii, 68, 69, 70

[58] D. Krishnan and R. Fergus. Fast image deconvolution using hyper-laplacian priors. In *NIPS 2009*. → pages 2, 10, 93, 97, 109

[59] D. Krishnan, T. Tay, and R. Fergus. Blind deconvolution using a normalized sparsity measure. In *CVPR 2011*. → pages 2, 10, 14, 60, 68, 70

[60] L. Lelégard, E. Delaygue, M. Brédif, and B. Vallet. Detecting and correcting motion blur from images shot with channel-dependent exposure time. In *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume 1, pages 341–346, 2012. → pages 43, 63

[61] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2:164–168, 1944. → pages 28

[62] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *CVPR 2009*, . → pages 50, 55

[63] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *CVPR 2009*, . → pages 87, 108

[64] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Efficient marginal likelihood optimization in blind deconvolution. In *CVPR 2011*, . → pages x, 108, 109, 120

[65] A. Levin, R. Fergus, F. Durand, and W. T. Freeman. Image and depth from a conventional camera with a coded aperture. 26(3):70, 2007. → pages 12

[66] A. Levin, R. Fergus, F. Durand, and W. T. Freeman. Image and depth from a conventional camera with a coded aperture. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 26(3), July 2007. → pages 49, 50

[67] A. Levin, R. Fergus, F. Durand, and W. T. Freeman. Deconvolution using natural image priors. *Massachusetts Institute of Technology, Computer Science and Artificial Intelligence Laboratory*, 2007. → pages 109

[68] A. Levin, R. Fergus, F. Durand, and W. T. Freeman. Image and depth from a conventional camera with a coded aperture. *ACM transactions on graphics*, 26(3):70, 2007. → pages 2, 10, 24, 25, 108, 109

[69] Y. Li and S. Osher. Coordinate descent optimization for l_1 minimization with application to compressed sensing; a greedy algorithm. *Inverse Problems and Imaging*, 3(3):487–503, 2009. → pages 50

[70] L. Lucy. An iterative technique for the rectification of observed distributions. *Astronomical Journal*, 79(2):745–754, 1974. → pages 14

[71] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *ICCV 2009*. → pages 1, 9, 10, 101

[72] C. A. Metzler, A. Maleki, and R. G. Baraniuk. Bm3d-amp: A new image recovery algorithm based on bm3d denoising. In *Image Processing (ICIP), 2015 IEEE International Conference on*, pages 3116–3120. IEEE, 2015. → pages 120

[73] T. Michaeli and M. Irani. Blind deblurring using internal patch recurrence. In *ECCV 2014*. → pages 14

[74] P. Milanfar. A tour of modern image filtering: New insights and methods, both practical and theoretical. *IEEE Signal Processing Magazine*, 30(1): 106–128, 2013. → pages 1

[75] J. Miskin and D. J. MacKay. Ensemble learning for blind image separation and deconvolution. In *NIPS 2000*. → pages 13

[76] J. Nocedal and S. J. Wright. Numerical optimization 2nd. 2006. → pages 28

[77] S. Osher and L. I. Rudin. Feature-oriented image enhancement using shock filters. *SIAM Journal on Numerical Analysis*, 27(4):pp. 919–940, 1990. → pages 14

[78] J. Pan, Z. Hu, Z. Su, and M.-H. Yang. Deblurring text images via l0-regularized intensity and gradient prior. In *CVPR 2014*. → pages 5, 15, 86, 87, 88, 89, 91

[79] N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):123–231, 2013. → pages 2, 6, 18, 19, 21

[80] J. Park, H. Kim, Y.-W. Tai, M. S. Brown, and I. Kweon. High quality depth map upsampling for 3d-tof cameras. In *ICCV 2011*. → pages 17

[81] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990. → pages 55

[82] J. Ragan-Kelley, C. Barnes, A. Adams, S. Paris, F. Durand, and S. Amarasinghe. Halide: a language and compiler for optimizing parallelism, locality, and recomputation in image processing pipelines. *ACM SIGPLAN Notices*, 48(6):519–530, 2013. → pages 108

[83] A. Rav-Acha and S. Peleg. Two motion-blurred images are better than one. *Pattern Recognition Letter*, 26(3):311–317, 2005. → pages 43

[84] W. H. Richardson. Bayesian-based iterative method of image restoration. *Journal of the Optical Society of America*, 62(1):55–59, 1972. → pages 14

[85] Y. Romano, M. Elad, and P. Milanfar. The little engine that could: Regularization by denoising (red). *arXiv preprint arXiv:1611.02862*, 2016. → pages 120

[86] S. Roth and M. Black. Fields of experts. *International Journal of Computer Vision*, 82(2):205–229, . → pages 2, 5, 10

[87] S. Roth and M. J. Black. Fields of experts: A framework for learning image priors. In *CVPR 2005*, . → pages x, 5, 78, 96, 101, 102, 108

[88] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1-4):259–268, 1992. → pages 1, 8, 12

[89] S. Sardy, A. Bruce, and P. Tseng. Block coordinate relaxation methods for nonparametric wavelet denoising. *Journal of computational and graphical statistics*, 9(2), 2000. → pages 50

[90] K. Schelten, S. Nowozin, J. Jancsary, C. Rother, and S. Roth. Interleaved regression tree field cascades for blind image deconvolution. In *WACV 2015*. → pages 14, 78, 83

[91] M. Schmidt. minfunc: unconstrained differentiable multivariate optimization in matlab. http://www.cs.ubc.ca/~schmidtm/Software/minFunc.html. → pages 85, 99

[92] U. Schmidt and S. Roth. Shrinkage fields for effective image restoration. In *CVPR 2014*. → pages 2, 5, 12, 13, 76, 78, 79, 81, 84, 85, 93, 95, 97, 99, 101, 108, 141, 142, 144

[93] U. Schmidt, C. Rother, S. Nowozin, J. Jancsary, and S. Roth. Discriminative non-blind deblurring. In *CVPR 2013*. → pages 13, 14, 108, 109

[94] C. J. Schuler, H. Christopher Burger, S. Harmeling, and B. Scholkopf. A machine learning approach for non-blind image deconvolution. In *CVPR 2013*. → pages 13, 109

[95] C. J. Schuler, M. Hirsch, S. Harmeling, and B. Schölkopf. Learning to deblur. *arXiv preprint arXiv:1406.7444*, 2014. → pages 14

[96] S. Schuon, C. Theobalt, J. Davis, and S. Thrun. Lidarboost: Depth superresolution for ToF 3D shape scanning. In *CVPR 2009*. → pages 17

[97] S. Shalev-Shwartz and A. Tewari. Stochastic methods for ℓ_1-regularized loss minimization. *The Journal of Machine Learning Research*, 12: 1865–1892, 2011. → pages 50

[98] Q. Shan, J. Jia, and A. Agarwala. High-quality motion deblurring from a single image. In *ACM Transactions on Graphics (Proc. SIGGRAPH)*, pages 73:1–73:10, 2008. → pages 60, 61, 68, 70

[99] L. Sun, S. Cho, J. Wang, and J. Hays. Edge-based blur kernel estimation using patch priors. In *ICCP 2013*. → pages 14

[100] Y.-W. Tai and S. Lin. Motion-aware noise filtering for deblurring of noisy and blurry images. In *CVPR 2012*. → pages 14

[101] H. Talebi and P. Milanfar. Global image denoising. *IEEE Transactions on Image Processing*, 23(2):755–768, 2014. → pages 1, 9

[102] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *ICCV 1998*. → pages 1, 8, 9, 14

[103] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg. Plug-and-play priors for model based reconstruction. In *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*, pages 945–948. IEEE, 2013. → pages 120

[104] R. Wang, Z. Yang, L. Liu, J. Deng, and F. Chen. Decoupling noise and features via weighted ℓ_1-analysis compressed sensing. *ACM Transactions on Graphics*, 33(2):18, 2014. → pages 25

[105] J. Weickert. *Anisotropic diffusion in image processing*. ECMI Series, Teubner-Verlag, Stuttgart, Germany, 1998. → pages 1, 8

[106] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce. Non-uniform deblurring for shaken images. In *CVPR 2010*. → pages 14, 68, 70

[107] N. Wiener. *Extrapolation, interpolation, and smoothing of stationary time series*, volume 7. MIT press Cambridge, MA, 1949. → pages 12

[108] B. Wohlberg. Efficient algorithms for convolutional sparse representations. *IEEE Transactions on Image Processing*, 25(1):301–315, 2016. → pages 10

[109] L. Xiao, F. Heide, W. Heidrich, B. Schölkopf, and M. Hirsch. Learning proximal operators for image restoration. In *Under review of ICCV 2017*, . → pages 6, 109

[110] L. Xiao, F. Heide, M. O'Toole, A. Kolb, M. B. Hullin, K. Kutulakos, and W. Heidrich. Defocus deblurring and superresolution for time-of-flight depth cameras. In *CVPR 2015*, . → pages 3

[111] L. Xiao, J. Wang, W. Heidrich, and M. Hirsch. Learning high-order filters for efficient blind deconvolution of document photographs. In *ECCV 2016*, . → pages 5

[112] L. Xiao, J. Gregson, F. Heide, and W. Heidrich. Stochastic blind motion deblurring. *IEEE Transactions on Image Processing*, 24(10):3071–3085, Oct 2015. → pages 4

[113] L. Xu and J. Jia. Two-phase kernel estimation for robust motion deblurring. In *ECCV 2010*. → pages 14, 60, 61, 68, 70, 79, 86

[114] L. Xu, S. Zheng, and J. Jia. Unnatural l0 sparse representation for natural image deblurring. In *CVPR 2013*. → pages 14, 60, 61, 68, 86

[115] T. Yue, S. Cho, J. Wang, and Q. Dai. Hybrid image deblurring by fusing edge and power spectrum information. In *ECCV 2014*. → pages 14

[116] L. Zhong, S. Cho, D. Metaxas, S. Paris, and J. Wang. Handling noise in single image deblurring using directional filters. In *CVPR 2013*. → pages 14

[117] J. Zhu, L. Wang, R. Yang, J. E. Davis, and Z. Pan. Reliability fusion of time-of-flight depth and stereo geometry for high quality depth maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(7): 1400–1414, 2011. → pages 17

[118] D. Zoran and Y. Weiss. From learning models of natural image patches to whole image restoration. In *ICCV 2011*. → pages 2, 5, 10, 11, 101, 109

[119] W. Zuo, D. Ren, S. Gu, L. Lin, and L. Zhang. Discriminative learning of iteration-wise priors for blind deconvolution. In *CVPR 2015*. → pages 14, 83

Appendix A

Supplemental Materials

A.1 Supplemental material for Chapter 3

This section provides implementation details for Algorithm 2 (updating amplitude estimate) and Algorithm 3 (updating depth estimate) in Chapter 3. The symbol ∇ defines the derivative operator, T defines the matrix transpose, and $\mathbf{I}$ defines the identity matrix.

Algorithm 2, Line 2:

$$\mathbf{a} = \underset{\mathbf{a}}{\operatorname{argmin}}\, \rho||\mathbf{c} - \mathbf{A}\mathbf{a}||_2^2 + \lambda_1 \rho_a ||\nabla \mathbf{a} - \mathbf{y} - \mathbf{p}_1 + \mathbf{u}_1||_2^2 \tag{A.1}$$

equals to the solution of the linear equation system:

$$(\rho \mathbf{A}^{\mathsf{T}} \mathbf{A} + \lambda_1 \rho_a \nabla^{\mathsf{T}} \nabla)\mathbf{a} = \rho \mathbf{A}^{\mathsf{T}} \mathbf{c} + \lambda_1 \rho_a \nabla^{\mathsf{T}} (\mathbf{y} + \mathbf{p}_1 - \mathbf{u}_1) \tag{A.2}$$

and we solve it by the left division function in Matlab.

Algorithm 2, Line 3:

$$\mathbf{y} = \underset{\mathbf{y}}{\operatorname{argmin}}\, \lambda_1 ||\nabla \mathbf{a} - \mathbf{y} - \mathbf{p}_1 + \mathbf{u}_1||_2^2 + \lambda_2 ||\nabla \mathbf{y} - \mathbf{p}_2 + \mathbf{u}_2||_2^2 \tag{A.3}$$

equals to the solution of the linear equation system:

$$(\lambda_1 \mathbf{I} + \lambda_2 \nabla^{\mathsf{T}} \nabla)\mathbf{y} = \lambda_1(\nabla \mathbf{a} - \mathbf{p}_1 + \mathbf{u}_1) + \lambda_2 \nabla^{\mathsf{T}}(\mathbf{p}_2 - \mathbf{u}_2) \tag{A.4}$$

and solved by the left division function in Matlab.

Algorithm 2, Line 4:

$$\mathbf{p}_1 = \underset{\mathbf{p}_1}{\operatorname{argmin}} ||\mathbf{p}_1||_1 + \rho_a||\nabla \mathbf{a} - \mathbf{y} - \mathbf{p}_1 + \mathbf{u}_1||_2^2 \tag{A.5}$$

is a soft shrinkage problem and has closed form solution:

$$\mathbf{p}_1 = \text{soft-shrinkage}(\nabla \mathbf{a} - \mathbf{y} + \mathbf{u}_1, \frac{0.5}{\rho_a}) \tag{A.6}$$

where the soft-shrinkage operator is defined as:

$$\text{soft-shrinkage}(\mathbf{x}, \epsilon) = \begin{cases} \mathbf{x} + \epsilon; \mathbf{x} < -\epsilon \\ \mathbf{0}; -\epsilon \leq \mathbf{x} \leq \epsilon \\ \mathbf{x} - \epsilon; \mathbf{x} > \epsilon \end{cases} \tag{A.7}$$

Algorithm 2, Line 5:

$$\mathbf{p}_2 = \underset{\mathbf{p}_2}{\operatorname{argmin}} ||\mathbf{p}_2||_1 + \rho_a||\nabla \mathbf{y} - \mathbf{p}_2 + \mathbf{u}_2||_2^2 \tag{A.8}$$

is a soft shrinkage problem and has closed form solution:

$$\mathbf{p}_2 = \text{soft-shrinkage}(\nabla \mathbf{y} + \mathbf{u}_2, \frac{0.5}{\rho_a}) \tag{A.9}$$

Algorithm 3, Line 2:

$$\mathbf{z} = \underset{\mathbf{z}}{\operatorname{argmin}} \overbrace{\rho||\mathbf{c} - \mathbf{a} \circ \mathbf{g}(\mathbf{z})||_2^2}^{\text{data fitting constraint}} + \overbrace{\tau_1 \rho_x ||\nabla \mathbf{z} - \mathbf{x} - \mathbf{q}_1 + \mathbf{v}_1||_2^2}^{\text{prior constraint}} \tag{A.10}$$

is a nonlinear least squares problem due to the nonlinearity of the modulation function $\mathbf{g}(\mathbf{z})$. We solve this problem by the Levenberg-Marquardt method implemented in the lsqnonlin(.) function in Matlab. We provide the analytical Jacobian for acceleration:

$$J(\mathbf{z}) = \begin{bmatrix} J_{data}(\mathbf{z}) \\ J_{prior} \end{bmatrix} \tag{A.11}$$

where the matrix $J_{data}(\mathbf{z})$ and J_{prior} define the Jacobian of the 1^{st} (data fitting constraint) and 2^{nd} (prior constraint) least squares in Eq.A.10 respectively.

Since the 1^{st} least squares are pixel-wise separable (benefit from our splitting method explained in Section 3.2.1 in Chapter 3), $J_{data}(\mathbf{z})$ is simply a diagonal matrix composed of:

$$-\mathbf{a}_k \cdot \frac{\partial \mathbf{g}(\mathbf{z}_k)}{\partial \mathbf{z}_k} \cdot \sqrt{\rho} \tag{A.12}$$

where k is the pixel index. For the ToF cameras based on cosine model modulation (see Eq.3.1 in Chapter 3), the diagonal element in Eq.A.12 becomes:

$$\mathbf{a}_k \cdot i\frac{4\pi f}{c} \cdot e^{-i(\frac{4\pi f}{c} \cdot \mathbf{z}_k)} \cdot \sqrt{\rho} \tag{A.13}$$

For arbitrary modulation waveforms in the future, the diagonal element in Eq.A.12 can be estimated from calibration data. J_{prior} is simply the matrix version of the derivative operator ∇ multiplied by $\sqrt{\tau_1 \rho_x}$, which is independent of $\mathbf{z}$.

Algorithm 3, Line 3:

$$\mathbf{x} = \operatorname*{argmin}_{\mathbf{x}} \tau_1 ||\nabla \mathbf{z} - \mathbf{x} - \mathbf{q}_1 + \mathbf{v}_1||_2^2 + \tau_2 ||\nabla \mathbf{x} - \mathbf{q}_2 + \mathbf{v}_2||_2^2 \tag{A.14}$$

equals to the solution of the linear equation system:

$$(\tau_1 \mathbf{I} + \tau_2 \nabla^{\mathsf{T}} \nabla)\mathbf{x} = \tau_1(\nabla \mathbf{z} - \mathbf{q}_1 + \mathbf{v}_1) + \tau_2 \nabla^{\mathsf{T}}(\mathbf{q}_2 - \mathbf{v}_2) \tag{A.15}$$

and solved by the left division function in Matlab.

Algorithm 3, Line 4:

$$\mathbf{q}_1 = \underset{\mathbf{q}_1}{\operatorname{argmin}} ||\mathbf{q}_1||_1 + \rho_x||\nabla\mathbf{z} - \mathbf{x} - \mathbf{q}_1 + \mathbf{v}_1||_2^2 \tag{A.16}$$

is a soft shrinkage problem and has closed form solution:

$$\mathbf{q}_1 = \text{soft-shrinkage}(\nabla\mathbf{z} - \mathbf{x} + \mathbf{v}_1, \frac{0.5}{\rho_x}) \tag{A.17}$$

Algorithm 3, Line 5:

$$\mathbf{q}_2 = \underset{\mathbf{q}_2}{\operatorname{argmin}} ||\mathbf{q}_2||_1 + \rho_x||\nabla\mathbf{x} - \mathbf{q}_2 + \mathbf{v}_2||_2^2 \tag{A.18}$$

is a soft shrinkage problem and has closed form solution:

$$\mathbf{q}_2 = \text{soft-shrinkage}(\nabla\mathbf{x} + \mathbf{v}_2, \frac{0.5}{\rho_x}) \tag{A.19}$$

A.2 Supplemental material for Chapter 5

This section gives the derivation details of Eq.5.6 -5.10 in Chapter 5.

Derivation of Eq.5.6 and5.7

The objective of latent image estimation at t-th iteration is defined as

$$\mathbf{x}^t = \underset{\mathbf{x}^t}{\operatorname{argmin}} ||\mathbf{b} - \mathbf{k}^{t-1} \otimes \mathbf{x}^t||_2^2 + \sum_{i=1}^{N} \rho_i^t(\mathbf{F}_i^t \mathbf{x}^t) \tag{A.20}$$

Applying the half-quadratic splitting [30] on each filter response $\mathbf{F}_i^t \mathbf{x}^t$, we have

$$\mathbf{x}^t = \underset{\mathbf{x}^t, \mathbf{u}_i^t}{\operatorname{argmin}} ||\mathbf{b} - \mathbf{k}^{t-1} \otimes \mathbf{x}^t||_2^2 + \sum_{i=1}^{N} \left(\rho_i^t(\mathbf{u}_i^t) + \frac{\beta^t}{2} ||\mathbf{F}_i^t \mathbf{x}^t - \mathbf{u}_i^t||_2^2 \right) \tag{A.21}$$

The half-quadratic optimization technique [30] solves Eq.A.21 by alternatively updating $\mathbf{x}^t$ and $\mathbf{u}_i^t$. More specifically, for $\mathbf{x}^t$ update,

$$\mathbf{x}^t = \underset{\mathbf{x}^t}{\operatorname{argmin}} ||\mathbf{b} - \mathbf{k}^{t-1} \otimes \mathbf{x}^t||_2^2 + \sum_{i=1}^{N} \frac{\beta^t}{2} ||\mathbf{F}_i^t \mathbf{x}^t - \mathbf{u}_i^t||_2^2 \tag{A.22}$$

This is solved by setting the gradient of the right-hand linear least squares to be zero,

$$\left(\mathbf{K}_{t-1}^{\mathsf{T}} \mathbf{K}_{t-1} + \frac{\beta^t}{2} \sum_{i=1}^{N} \mathbf{F}_i^{t\,\mathsf{T}} \mathbf{F}_i^t \right) \mathbf{x}^t = \mathbf{K}_{t-1}^{\mathsf{T}} \mathbf{b} + \frac{\beta^t}{2} \sum_{i=1}^{N} \mathbf{F}_i^{t\,\mathsf{T}} \mathbf{u}_i^t \tag{A.23}$$

where matrix $\mathbf{K}_{t-1}$ represents corresponding convolution with blur kernel $\mathbf{k}^{t-1}$. Because $\mathbf{K}_{t-1}$ and $\mathbf{F}_i^t$ represent convolution process, Eq.A.23 can be efficiently solved in Fourier domain. Let $\lambda^t = \beta^t/2$, we have

$$\mathbf{x}^t = \mathcal{F}^{-1} \left[\frac{\mathcal{F}(\mathbf{K}_{t-1}^{\mathsf{T}} \mathbf{b} + \lambda^t \sum_{i=1}^{N} \mathbf{F}_i^{t\,\mathsf{T}} \mathbf{u}_i^t)}{\mathcal{F}(\mathbf{K}_{t-1}^{\mathsf{T}}) \cdot \mathcal{F}(\mathbf{K}_{t-1}) + \lambda^t \sum_{i=1}^{N} \mathcal{F}(\mathbf{F}_i^{t\,\mathsf{T}}) \cdot \mathcal{F}(\mathbf{F}_i^t)} \right] \tag{A.24}$$

For $\mathbf{u}_i^t$ update,

$$\mathbf{u}_i^t = \underset{\mathbf{u}_i^t}{\operatorname{argmin}}\, \rho_i^t(\mathbf{u}_i^t) + \frac{\beta^t}{2}||\mathbf{F}_i^t \mathbf{x}^{t-1} - \mathbf{u}_i^t||_2^2 \tag{A.25}$$

This minimization problem is pixel-wise separable. The key idea of [92] is to replace this minimization problem with a shrinkage function ψ_i^t, i.e.,

$$\mathbf{u}_i^t = \psi_i^t(\mathbf{F}_i^t \mathbf{x}^{t-1}) \tag{A.26}$$

By Substituting Eq.A.26 to Eq.A.24, we have

$$\mathbf{x}^t = \mathcal{F}^{-1}\left[\frac{\mathcal{F}(\mathbf{K}_{t-1}^\mathsf{T}\mathbf{b} + \lambda^t \sum_{i=1}^N {\mathbf{F}_i^t}^\mathsf{T} \psi_i^t(\mathbf{F}_i^t \mathbf{x}^{t-1}))}{\mathcal{F}(\mathbf{K}_{t-1}^\mathsf{T}) \cdot \mathcal{F}(\mathbf{K}_{t-1}) + \lambda^t \sum_{i=1}^N \mathcal{F}({\mathbf{F}_i^t}^\mathsf{T}) \cdot \mathcal{F}(\mathbf{F}_i^t)}\right] \tag{A.27}$$

Note that we replace $\mathbf{x}^{t-1}$ with $\mathbf{z}^{t-1}$ in our blind deblurring framework. Therefore, Eq.A.27 becomes

$$\mathbf{x}^t = \mathcal{F}^{-1}\left[\frac{\mathcal{F}(\mathbf{K}_{t-1}^\mathsf{T}\mathbf{b} + \lambda^t \sum_{i=1}^N {\mathbf{F}_i^t}^\mathsf{T} \psi_i^t(\mathbf{F}_i^t \mathbf{z}^{t-1}))}{\mathcal{F}(\mathbf{K}_{t-1}^\mathsf{T}) \cdot \mathcal{F}(\mathbf{K}_{t-1}) + \lambda^t \sum_{i=1}^N \mathcal{F}({\mathbf{F}_i^t}^\mathsf{T}) \cdot \mathcal{F}(\mathbf{F}_i^t)}\right] \tag{A.28}$$

which is Eq.5.6 in Chapter 5. Eq.5.7 in Chapter 5 can be derived in similar way thus it's omitted here.

Derivation of Eq.5.8

The objective of kernel estimation is defined as

$$\mathbf{k}^t = \underset{\mathbf{k}^t}{\operatorname{argmin}}\, ||\mathbf{b} - \mathbf{k}^t \otimes \mathbf{z}^t||_2^2 + \tau^t ||\mathbf{k}^t||_2^2 \tag{A.29}$$

This is solved by setting the gradient of the right-hand linear least squares to be zero,

$$(\mathbf{Z}_t^\mathsf{T}\mathbf{Z}_t + \tau^t)\mathbf{k}^t = \mathbf{Z}_t^\mathsf{T}\mathbf{b} \tag{A.30}$$

where matrix $\mathbf{Z}_t$ represents corresponding convolution with image $\mathbf{z}^t$. Eq.A.30 can be efficiently solved in Fourier domain:

$$\mathbf{k}^t = \mathcal{F}^{-1}\left[\frac{\mathcal{F}(\mathbf{z}^t)^* \cdot \mathcal{F}(\mathbf{b})}{\mathcal{F}(\mathbf{z}^t)^* \cdot \mathcal{F}(\mathbf{z}^t) + \tau^t}\right] \tag{A.31}$$

which is Eq.5.8 in the main paper. $*$ represents conjugate transpose.

Calculation of Eq.5.9

For $\mathbf{x}^t$ update, the training loss $\ell = ||\mathbf{x}^t - \bar{\mathbf{x}}||_2^2$. Its gradient w.r.t. the model parameters $\Theta^t = (\mathbf{f}_i^t, \psi_i^t, \lambda^t)$ is computed as

$$\frac{\partial \ell}{\Theta^t} = \frac{\partial \mathbf{x}^t}{\partial \Theta^t}\frac{\partial \ell}{\mathbf{x}^t} = 2\frac{\partial \mathbf{x}^t}{\partial \Theta^t}(\mathbf{x}^t - \bar{\mathbf{x}}) \tag{A.32}$$

In order to compute $\partial \mathbf{x}^t / \partial \Theta^t$, $\mathbf{x}^t$ in Eq.A.28 can be rewritten as

$$\mathbf{x}^t = \left(\mathbf{K}_{t-1}^\mathsf{T}\mathbf{K}_{t-1} + \lambda^t \sum_{i=1}^N {\mathbf{F}_i^t}^\mathsf{T}\mathbf{F}_i^t\right)^{-1}\left(\mathbf{K}_{t-1}^\mathsf{T}\mathbf{b} + \lambda^t \sum_{i=1}^N {\mathbf{F}_i^t}^\mathsf{T}\psi_i^t(\mathbf{F}_i^t\mathbf{z}^{t-1})\right) \tag{A.33}$$

Then $\partial \mathbf{x}^t / \partial \Theta^t$ can be derived following the matrix calculus rules, and we refer the derivation details of $\partial \mathbf{x}^t / \partial \Theta^t$ to the supplemental material of [92].

Calculation of Eq.5.10

For $\mathbf{z}^t$ and $\mathbf{k}^t$ update, the training loss $\ell = ||\mathbf{k}^t - \bar{\mathbf{k}}||_2^2 + \alpha||\mathbf{z}^t - \bar{\mathbf{x}}||_2^2$. Its gradient w.r.t.the model parameters $\Omega^t = (\mathbf{g}_i^t, \phi_i^t, \eta^t, \tau^t)$ is computed as

$$\frac{\partial \ell}{\partial \Omega^t} = \frac{\partial \mathbf{z}^t}{\partial \Omega^t}\frac{\partial \mathbf{k}^t}{\partial \mathbf{z}^t}\frac{\partial \ell}{\partial \mathbf{k}^t} + \frac{\partial \mathbf{k}^t}{\partial \Omega^t}\frac{\partial \ell}{\partial \mathbf{k}^t} + \frac{\partial \mathbf{z}^t}{\partial \Omega^t}\frac{\partial \ell}{\partial \mathbf{z}^t} \tag{A.34}$$

where

$$\frac{\partial \ell}{\partial \mathbf{k}^t} = 2(\mathbf{k}^t - \bar{\mathbf{k}}) \tag{A.35}$$

$$\frac{\partial \ell}{\partial \mathbf{z}^t} = 2\alpha(\mathbf{z}^t - \bar{\mathbf{x}}) \tag{A.36}$$

In order to compute $\partial \mathbf{k}^t / \partial \mathbf{z}^t$ and $\partial \mathbf{k}^t / \partial \Omega^t$, $\mathbf{k}^t$ in Eq.A.31 can be rewritten as

$$\mathbf{k}^t = (\mathbf{Z}_t^\mathsf{T} \mathbf{Z}_t + \tau^t)^{-1} (\mathbf{Z}_t^\mathsf{T} \mathbf{b}) \tag{A.37}$$

For brevity, we denote $(\mathbf{Z}_t^\mathsf{T} \mathbf{Z}_t + \tau^t)$ as $\mathbf{\Pi}$, and $(\mathbf{Z}_t^\mathsf{T} \mathbf{b})$ as $\mathbf{\Lambda}$, then $\mathbf{k}^t = \mathbf{\Pi}^{-1} \mathbf{\Lambda}$. We use the square bracket around any image $\mathbf{a}$, i.e. $[\mathbf{a}]$, to indicate the matrix representing convolution with $\mathbf{a}$. Moreover, we define matrix $\mathcal{R}$ representing rotating a 2D image by 180 degrees, i.e., $\mathcal{R}\mathbf{a}$ means rotating the 2D image $\mathbf{a}$ by 180 degrees. Then, we have

$$\begin{aligned}
\frac{\partial \mathbf{k}^t}{\partial \mathbf{z}^t} &= \frac{\partial (\mathbf{\Pi}^{-1} \mathbf{\Lambda})}{\partial \mathbf{z}^t} \\
&= \frac{\partial \mathbf{\Lambda}}{\partial \mathbf{z}^t} \mathbf{\Pi}^{-1} - \mathbf{\Lambda}^\mathsf{T} \mathbf{\Pi}^{-1} \frac{\partial \mathbf{\Pi}}{\partial \mathbf{z}^t} \mathbf{\Pi}^{-1} \\
&= \frac{\partial (\mathbf{Z}_t^\mathsf{T} \mathbf{b})}{\partial \mathbf{z}^t} \mathbf{\Pi}^{-1} - \frac{\partial (\mathbf{\Pi} \mathbf{k}^t)}{\partial \mathbf{z}^t} \mathbf{\Pi}^{-1} \\
&= \frac{\partial ([\mathbf{b}] \mathcal{R} \mathbf{z}^t)}{\partial \mathbf{z}^t} \mathbf{\Pi}^{-1} - \frac{\partial ((\mathbf{Z}_t^\mathsf{T} \mathbf{Z}_t + \tau^t) \mathbf{k}^t)}{\partial \mathbf{z}^t} \mathbf{\Pi}^{-1} \\
&= \mathcal{R}^\mathsf{T} [\mathbf{b}]^\mathsf{T} \mathbf{\Pi}^{-1} - \frac{\partial (\mathbf{Z}_t^\mathsf{T} \mathbf{Z}_t \mathbf{k}^t)}{\partial \mathbf{z}^t} \mathbf{\Pi}^{-1} \\
&= \mathcal{R}^\mathsf{T} [\mathbf{b}]^\mathsf{T} \mathbf{\Pi}^{-1} - (\mathcal{R}^\mathsf{T} [\mathbf{Z}_t \mathbf{k}^t]^\mathsf{T} + \mathbf{K}_t^\mathsf{T} \mathbf{Z}_t) \mathbf{\Pi}^{-1} \\
&= (\mathcal{R}^\mathsf{T} [\mathbf{b}]^\mathsf{T} - \mathcal{R}^\mathsf{T} [\mathbf{Z}_t \mathbf{k}^t]^\mathsf{T} - \mathbf{K}_t^\mathsf{T} \mathbf{Z}_t) \mathbf{\Pi}^{-1}
\end{aligned} \tag{A.38}$$

And,

$$\begin{aligned}
\frac{\partial \mathbf{k}^t}{\partial \Omega^t} &= \frac{\partial \mathbf{\Pi}^{-1} \mathbf{\Lambda}}{\partial \tau^t} \\
&= \frac{\partial \mathbf{\Lambda}}{\partial \tau^t} \mathbf{\Pi}^{-1} - \mathbf{\Lambda}^\mathsf{T} \mathbf{\Pi}^{-1} \frac{\partial \mathbf{\Pi}}{\partial \tau^t} \mathbf{\Pi}^{-1} \\
&= -\mathbf{\Lambda}^\mathsf{T} \mathbf{\Pi}^{-1} \frac{\partial (\mathbf{Z}_t^\mathsf{T} \mathbf{Z}_t + \tau^t)}{\partial \tau^t} \mathbf{\Pi}^{-1} \\
&= -\mathbf{K}_t^\mathsf{T} \mathbf{\Pi}^{-1}
\end{aligned} \tag{A.39}$$

By substituting Eq.A.35, A.36, A.38, A.39 into Eq.A.34, we have

$$\begin{aligned}
\frac{\partial \ell}{\partial \Omega^t} &= -2\mathbf{K}_t^{\mathsf{T}}\mathbf{\Pi}^{-1}(\mathbf{k}^t - \bar{\mathbf{k}}) \\
&+ 2\frac{\partial \mathbf{z}^t}{\partial \Omega^t}\Big((\mathcal{R}^{\mathsf{T}}[\mathbf{b}]^{\mathsf{T}} - \mathcal{R}^{\mathsf{T}}[\mathbf{Z}_t\mathbf{k}^t]^{\mathsf{T}} - \mathbf{K}_t^{\mathsf{T}}\mathbf{Z}_t)\mathbf{\Pi}^{-1}(\mathbf{k}^t - \bar{\mathbf{k}}) + \alpha(\mathbf{z}^t - \bar{\mathbf{x}})\Big)
\end{aligned} \tag{A.40}$$

The derivation of $\partial\mathbf{z}^t/\partial\Omega^t$ is similar as $\partial\mathbf{x}^t/\partial\Theta^t$, and we refer the details to the supplemental material of [92]. We will make our training code publicly available with the paper.

A.3 Supplemental material for Chapter 6

This section gives the derivation of the analytic gradients that are required for training in Chapter 6.

The HQS iterations ($t = 1, 2, ..., T$) in our method read as follows:

$$\begin{aligned}
\mathbf{z}^t &= \mathbf{prox}_\Theta(\mathbf{x}^{t-1}) \\
\mathbf{x}^t &= \underbrace{\left(\lambda \mathbf{A}^\mathsf{T}\mathbf{A} + \rho^t\right)^{-1}}_{\mathbf{\Pi}_t} \underbrace{\left(\lambda \mathbf{A}^\mathsf{T}\mathbf{b} + \rho^t \mathbf{z}^t\right)}_{\mathbf{\Lambda}_t}
\end{aligned} \tag{A.41}$$

For both denoising and deconvolution tasks, the $\mathbf{x}^t$-update in Eq.A.41 has a closed-form solution, which is given in Eq.6.8 and 6.9 in Chapter 6. In this supplemental document we present a derivation with the more general formula $\mathbf{x}^t = \mathbf{\Pi}_t^{-1}\mathbf{\Lambda}_t$.

In our method, the trainable parameters $\Omega = \{\lambda_p, \Theta\}$ include the fidelity weight λ_p for each problem class p, and the model parameters Θ of the proximal fields that are shared across all problem classes. The training loss function ℓ is defined as the negative of the average PSNR between the reconstructed and ground truth images. The gradient of the loss ℓ w.r.t. Θ is computed by averaging the gradients of all images, while the gradient of the loss ℓ w.r.t. λ_p is computed by averaging the gradients of only those images that belong to class p. For convenience, we give the derivations for one image, and omit the class label p below.

$$\ell = -20 \log_{10} \left(\frac{255\sqrt{M}}{||\mathbf{x}^T - \mathbf{x}_{\text{true}}||_2} \right), \tag{A.42}$$

where M is the number of pixels in each image, $\mathbf{x}_{\text{true}}$ is the ground truth image, and $\mathbf{x}^T$ is the reconstructed image.

$$\begin{aligned}
\frac{\partial \ell}{\partial \Omega} &= \sum_{t=1}^{T} \left(\frac{\partial \mathbf{x}^t}{\partial \lambda} \frac{\partial \ell}{\partial \mathbf{x}^t} + \frac{\partial \mathbf{z}^t}{\partial \Theta} \left(\frac{\partial \ell}{\partial \mathbf{z}^t} + \frac{\partial \mathbf{x}^t}{\partial \mathbf{z}^t} \frac{\partial \ell}{\partial \mathbf{x}^t} \right) \right) \\
&= \sum_{t=1}^{T} \left(\frac{\partial \mathbf{x}^t}{\partial \lambda} + \frac{\partial \mathbf{z}^t}{\partial \Theta} \frac{\partial \mathbf{x}^t}{\partial \mathbf{z}^t} \right) \frac{\partial \ell}{\partial \mathbf{x}^t}
\end{aligned} \tag{A.43}$$

whereby we used $\partial \ell / \partial \mathbf{z}^t = 0$. Next, we provide the derivation for the partial derivative terms in Eq.A.43:

$$\begin{aligned}\frac{\partial \mathbf{x}^t}{\partial \lambda} &= \frac{\partial \mathbf{\Lambda}_t}{\partial \lambda}\mathbf{\Pi}_t^{-1} - \left(\mathbf{\Pi}_t^{-1}\mathbf{\Lambda}_t\right)^{\mathsf{T}}\frac{\partial \mathbf{\Pi}_t}{\partial \lambda}\mathbf{\Pi}_t^{-1} = \left(\frac{\partial \mathbf{\Lambda}_t}{\partial \lambda} - \frac{\partial \mathbf{\Pi}_t \mathbf{x}^t}{\partial \lambda}\right)\mathbf{\Pi}_t^{-1} \\ &= \left(\mathbf{A}^{\mathsf{T}}\mathbf{b} - \mathbf{A}^{\mathsf{T}}\mathbf{A}\mathbf{x}^t\right)^{\mathsf{T}}\mathbf{\Pi}_t^{-1}\end{aligned} \tag{A.44}$$

$$\frac{\partial \mathbf{x}^t}{\partial \mathbf{z}^t} = \frac{\partial \mathbf{\Lambda}_t}{\partial \mathbf{z}^t}\mathbf{\Pi}_t^{-1} - \left(\mathbf{\Pi}_t^{-1}\mathbf{\Lambda}_t\right)^{\mathsf{T}}\frac{\partial \mathbf{\Pi}_t}{\partial \mathbf{z}^t}\mathbf{\Pi}_t^{-1} = \frac{\partial \mathbf{\Lambda}_t}{\partial \mathbf{z}^t}\mathbf{\Pi}_t^{-1} = \rho^t\mathbf{\Pi}_t^{-1} \tag{A.45}$$

$$\frac{\partial \ell}{\partial \mathbf{x}^{t-1}} = \frac{\partial \mathbf{x}^t}{\partial \mathbf{x}^{t-1}}\frac{\partial \ell}{\partial \mathbf{x}^t} = \frac{\partial \mathbf{z}^t}{\partial \mathbf{x}^{t-1}}\frac{\partial \mathbf{x}^t}{\partial \mathbf{z}^t}\frac{\partial \ell}{\partial \mathbf{x}^t} = \rho^t\frac{\partial \mathbf{z}^t}{\partial \mathbf{x}^{t-1}}\mathbf{\Pi}_t^{-1}\frac{\partial \ell}{\partial \mathbf{x}^t} \tag{A.46}$$

To compute the gradient of $\mathbf{z}^t$ w.r.t. $\mathbf{x}^{t-1}$ and Θ, i.e. $\partial \mathbf{z}^t / \partial \mathbf{x}^{t-1}$ and $\partial \mathbf{z}^t / \partial \Theta$, let's first recall Eq. 6 in the main paper:

$$\mathbf{z}_k^t = \mathbf{z}_{k-1}^t - \sum_{i=1}^{N}\mathbf{F}_i^{k\mathsf{T}}\psi_i^k(\mathbf{F}_i^k\mathbf{z}_{k-1}^t), \quad s.t. \quad \mathbf{z}_0^t = \mathbf{x}^{t-1}, \quad k = 1,2,...,K \tag{A.47}$$

Note that $\mathbf{x}^{t-1}$ only appears at the first stage $k = 1$:

$$\mathbf{z}_1^t = \mathbf{z}_0^t - \sum_{i=1}^{N}\mathbf{F}_i^{1\mathsf{T}}\psi_i^1(\mathbf{F}_i^1\mathbf{z}_0^t) = \mathbf{x}^{t-1} - \sum_{i=1}^{N}\mathbf{F}_i^{1\mathsf{T}}\psi_i^1(\mathbf{F}_i^1\mathbf{x}^{t-1}) \tag{A.48}$$

Therefore,

$$\frac{\partial \mathbf{z}^t}{\partial \mathbf{x}^{t-1}} = \frac{\partial \mathbf{z}_K^t}{\partial \mathbf{x}^{t-1}} = \frac{\partial \mathbf{z}_1^t}{\partial \mathbf{x}^{t-1}}\frac{\partial \mathbf{z}_K^t}{\partial \mathbf{z}_1^t} = \left(\mathbf{I} - \sum_{i=1}^{N}\mathbf{F}_i^{1\mathsf{T}}\psi_i'^1(\mathbf{F}_i^1\mathbf{x}^{t-1})\mathbf{F}_i^1\right)\frac{\partial \mathbf{z}_K^t}{\partial \mathbf{z}_1^t} \tag{A.49}$$

where $\mathbf{I}$ is an identity matrix, and $\partial \mathbf{z}_K^t / \partial \mathbf{z}_1^t$ can be computed by following the rule below:

$$\frac{\partial \mathbf{z}_k^t}{\partial \mathbf{z}_{k-1}^t} = \mathbf{I} - \sum_{i=1}^{N}\mathbf{F}_i^{k\mathsf{T}}\psi_i'^k(\mathbf{F}_i^k\mathbf{z}_{k-1}^t)\mathbf{F}_i^k, \quad k = 1,2,...,K \tag{A.50}$$

$\partial \mathbf{z}^t / \partial \Theta$ in Eq.A.43 is composed of $\partial \mathbf{z}^t / \partial \mathbf{f}_i^k$ and $\partial \mathbf{z}^t / \partial \psi_i^k$, which are computed as follows:

$$\frac{\partial \mathbf{z}^t}{\partial \mathbf{f}_i^k} = \frac{\partial \mathbf{z}_k^t}{\partial \mathbf{f}_i^k}\frac{\partial \mathbf{z}_K^t}{\partial \mathbf{z}_k^t} = -\frac{\partial {\mathbf{F}_i^k}^{\mathsf{T}} \psi_i^k(\mathbf{F}_i^k \mathbf{z}_{k-1}^t)}{\partial \mathbf{f}_i^k}\frac{\partial \mathbf{z}_K^t}{\partial \mathbf{z}_k^t} \tag{A.51}$$

$$\frac{\partial \mathbf{z}^t}{\partial \psi_i^k} = \frac{\partial \mathbf{z}_k^t}{\partial \psi_i^k}\frac{\partial \mathbf{z}_K^t}{\partial \mathbf{z}_k^t} = -\frac{\partial {\mathbf{F}_i^k}^{\mathsf{T}} \psi_i^k(\mathbf{F}_i^k \mathbf{z}_{k-1}^t)}{\partial \psi_i^k}\frac{\partial \mathbf{z}_K^t}{\partial \mathbf{z}_k^t} \tag{A.52}$$